MAURÍCIO CARUZO REIS

CONSTRUA SEU COMPUTADOR POR MEIO SALÁRIO MÍNIMO

Petit
editora

CONSTRUA SEU COMPUTADOR POR MEIO SALÁRIO MÍNIMO

MAURÍCIO CARUZO REIS

PETIT
Editora e Marketing Direto Ltda.
Av. Brig. Luiz Antonio, 383 – 2º
São Paulo – Capital
Cep 01317

TODOS OS DIREITOS RESERVADOS

É expressamente proibida a reprodução parcial ou integral sem a autorização do editor

1986

ÍNDICE

INTRODUÇÃO 9

PARTE A — TEORIA DE ESTÁGIOS

Capítulo 1 — Memória

1. Tratamentos eletrônico de dados 11
2. Memória 18
3. RAM 6116 22
4. Compatibilidade RAM 6116 X EPROM 2716 25
5. Circuito de habilitação da memória 27

Capítulo 2 — Microprocessador

1. Histórico 29
2. O microprocessador em ação 30
3. Arquitetura de microprocessador 32
4. Organização interna do Z-80 34
5. Pinagem do Z-80 38
6. Temporização do Z-80 42

Capítulo 3 — Entrada de Dados

1. Esquema de bloco do estágio de entrada de dados 54
2. Operação DMA 59
3. Operação INPUT 65

Capítulo 4 — Saída de Dados

1. Esquema da saída de dados 66
2. Habilitação da saída de dados 69
3. Latch 74LS373 69
4. Apresentação de dados 70

Capítulo 5 — Circuitos Auxiliares

1. Circuito clock x passo-a-passo 71
2. Reset 73
3. Terminais desativados do microprocessador 74
4. Ponta de prova lógica 75
5. Fonte regulada 77

Capítulo 6 — Esquema Elétrico
1. Características elétricas 78
2. Expansão de memória 81
3. Expansão de porta de saída 81

Capítulo 7 — Descrição Física do Micro Galena
1. Blocos do Micro Galena 84
2. Painel 85
3. Ponta de prova lógica 90
4. Fonte de alimentação 91

PARTE B — MONTAGEM

Capítulo 8 — Montagem
1. Roteiro de montagem 94
2. Preparação da placa 96
3. Instalação dos soquetes de CIs 99
4. Colocação das chaves 101
5. Fiação 101
6. Lista de material 106

Capítulo 9 — Debugging (Teste de Montagem)
1. Dispositivos para debugging 110
2. Roteiro de teste de montagem 111
3. Teste do passo a passo 118
4. Teste de mau contato 119
5. Calibração do clock 119

PARTE C — PROGRAMAÇÃO

Capítulo 10 — Implementação do Assembler/Código de Máquinas
1. Programando o micro Galena 121

2. Leitura de memória 124
3. Entrada DMA simples e saída de dados 124
4. Transferência de dados entre registros 128
5. Transferência de dados entre pares de registros 133
6. Entrada e saída selecionada pelo registro C 140
7. Carga e transferência com os registros IX-IY 142
8. Carga e transferência com o ponteiro de pilha - SP 147
9. Interrupções 149
10. Entrada e saída de blocos de dados 153
11. Transferência de bloco de dados 155
12. Troca entre conteúdo de registros 158
13. Set e Reset de bit no registro 162
14. Operações com bit de flags 164
15. Instruções de desvio (jump) 166
16. Instruções de comparação 171
17. Instruções lógicas 173
18. Complementação do acumulador 176
19. Incrementação e decrementação de registro 179
20. Operações aritméticas 181
21. DAA — Aritmética de binários convertidos para BCD ... 185
22. Instruções de rotação e deslocamento 189
23. Sub-rotinas 198
24. Instruções de controle 201

Capítulo 11 — Programa Monitor

1. Carga com eco de programas extensos 202
2. Fluxogramas 206
3. Esquema de programa monitor 206
4. Esboço do jogo "Batalha Naval" 209

APÊNDICE

LISTA DE INSTRUÇÕES DO Z-80/FLAGS 223

INTRODUÇÃO

Quem já montou um radinho GALENA sabe que não existe neste mundo som mais lindo que seu chiado. Na verdade esse som é apenas o fundo musical do prazer de fazer as coisas, saber como funcionam seus mistérios e milagres. Menos sentimentalmente, a oportunidade de aprender, testar e trazer para o dia-a-dia as complicadas e distantes fórmulas teóricas.

A evolução da Eletrônica está acabando com esse prazer e essa oportunidade. Com os aparelhos extremamente complexos e fabricados por processos e instrumentos mais complexos ainda, não há chance para experiências de fundo de quintal, como o radinho GALENA.

Esta foi minha conclusão nos vários projetos de "construa seu próprio computador" de que participei. Além do grande número de serviços que devem ser contratados a terceiros (fotolitos, placa de circuito impresso, gravação da ROM), fica a frustante sensação de apenas estar montando um equipamento projetado em Nova York ou Tokio, sem a mínima possibilidade de criar ou melhorar qualquer coisa. Nestas circunstâncias é limitadíssimo o prazer de fazer e a oportunidade de aprender.

MICRO-GALENA, o computador aqui proposto, parte de uma concepção completamente diferente. Sua estrutura é totalmente aberta, tanto no hardware (parte eletrônica) quanto no software (programação). Isso deixa ao usuário uma ampla margem para experiências próprias, retirando-o da posição de simples montador.

No hardware se destaca, além do baixo custo e emprego somente de componentes facilmente encontráveis no mercado nacional, a possibilidade de expansões não padronizadas e facilidades para testes.

No software foi adotado o monitor programável - o próprio usuário programa o sistema que controla o computador, podendo reprogramá-lo a qualquer momento. Deste modo o MICRO-GALENA torna-se um eficiente instrumento não só de aprendizagem como também de testes.

O processo de montagem foi cuidadosamente planejado para ser executável no fundo do quintal: nada de circuito impresso ou gravação do ROM, utilização apenas de ferramentas caseiras.

Por suas características o MICRO-GALENA não se enquadra em qualquer categoria de micros-profissionais, domésticos ou de avaliação. Mais corretamente o MICRO-GALENA é um computador de cabeceira para quem quer conhecer e praticar computadores além de teclados reluzentes e vídeos brilhantes.

Maurício Caruzo Reis

PARTE A-TEORIA DE ESTAGIOS

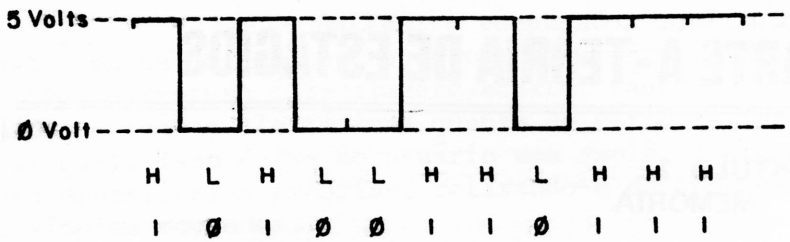
CAPÍTULO 1
MEMÓRIA

É na memória do computador que se armazenam os dados do programa. O presente capítulo discutirá o que são esses dados e quais as técnicas e componentes para armazená-los.

1 - TRATAMENTO ELETRÔNICO DE DADOS

Computadores trabalham com sinais elétricos (voltagem e corrente), e este é o único modo de informação que eles conseguem "entender" e manipular. Assim, se o usuário deseja passar para o computador uma palavra ela deve ser codificada em pulsos elétricos para depois ser processada.

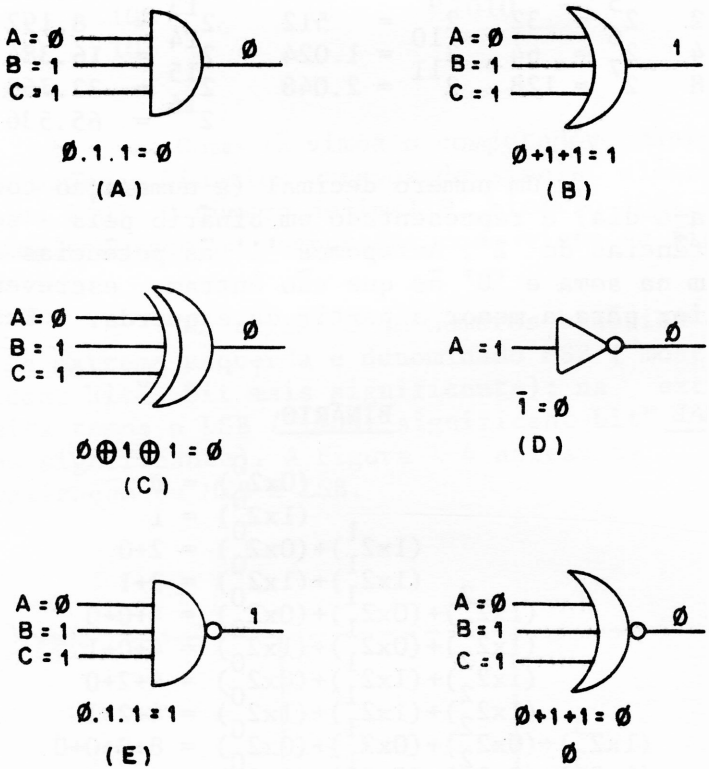
Quando a máquina precisa distinguir apenas dois tipos de sinais elétricos os circuitos ficam muito simples e precisos. Por esse motivo os computadores adotaram a Eletrônica Digital que emprega o sinal elétrico em dois estados (figura 1-1): (1) BIT H (também dito BIT 1) com voltagem maior - e certa especificação de corrente; (2) BIT L (também dito BIT 0) com voltagem menor - e certa especificação de corrente. Se o dispositivo encontra-se desligado do circuito e não apresenta nenhum dos estados H ou L dizemos que ele está em TRISTATE. Um conjunto de 8 bits é denominado BYTE ("báite").



1-1 sinal digital

Os bits são combinados em funções lógicas de acordo com as regras da Lógica Bivalente, o que na prática eletrônica é implementado através de PORTAS LÓGICAS. Na porta AND (figura 1-2A) a saída terá nível 0 se uma das entradas tiver nível 0, caso contrário será 1. Na porta OR (figura 1-2B) a saída será 1 se uma das entradas for 1, caso contrário será 0. Na porta XOR (também conhecida por EOR, figura 1-2C) a saída será 1 se houver número ímpar de entradas com 1. A porta NOT (ou INVERSORA, figura 1-2D) inverte na saída o sinal da entrada. A porta NAND (NOT + AND, figura 1-2E) é uma AND com saída invertida, ou seja, se uma das entradas for 0 a saída será 1. Na porta NOR (NOT + OR, figura 1-2F) temos uma OR invertida, se uma das entradas for 1 a saída será 0.

Portanto, o computador é uma máquina que trabalha exclusivamente com '1' e '0'; os outros números (de 2 a 9) são desconhecidos por ele. Torna-se, assim, necessário usar um sistema matemático baseado em apenas dois números - os NÚMEROS BINÁRIOS - que originam o que poderíamos chamar "Aritmética Digital".



1-2 portas lógicas (A) AND (B) OR (C) XOR (D) NOT (E) NAND (F) NOR

Os números binários são baseados em potências de '2', cuja tabela é apresentada a seguir (note bem que qualquer número elevado a potência 0 é igual a 1; daí $2^0=1$):

$2^0 = 1$	$2^4 = 16$	$2^8 = 256$	$2^{12} = 4.096$
$2^1 = 2$	$2^5 = 32$	$2^9 = 512$	$2^{13} = 8.192$
$2^2 = 4$	$2^6 = 64$	$2^{10} = 1.024$	$2^{14} = 16.384$
$2^3 = 8$	$2^7 = 128$	$2^{11} = 2.048$	$2^{15} = 32.768$
			$2^{16} = 65.536$

Um número decimal (a numeração comum do dia-a-dia) é representado em binário pela soma de potências de '2'. Antepomos '1' às potências que entram na soma e '0' às que não entram, escrevendo da maior para a menor a partir da esquerda. Exemplos:

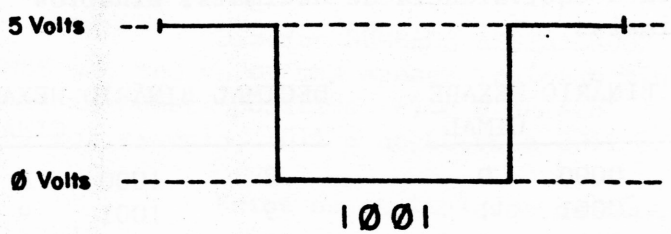
DECIMAL	BINÁRIO
0	$(0x2^0) = 0$
1	$(1x2^0) = 1$
2	$(1x2^1)+(0x2^0) = 2+0$
3	$(1x2^1)+(1x2^0) = 2+1$
4	$(1x2^2)+(0x2^1)+(0x2^0) = 4+0+0$
5	$(1x2^2)+(0x2^1)+(1x2^0) = 4+0+1$
6	$(1x2^2)+(1x2^1)+(0x2^0) = 4+2+0$
7	$(1x2^2)+(1x2^1)+(1x2^0) = 4+2+1$
8	$(1x2^3)+(0x2^2)+(0x2^1)+(0x2^0) = 8+0+0+0$
9	$(1x2^3)+(0x2^2)+(0x2^1)+(1x2^0) = 8+0+0+1$
10	$(1x2^3)+(0x2^2)+(1x2^1)+(0x2^0) = 8+0+2+0$

As potências de '2' podem ser omitidas, subentendendo-as pela posição no número binário, e registrados apenas os '1' e '0'. Os exemplos anteriores ficam:

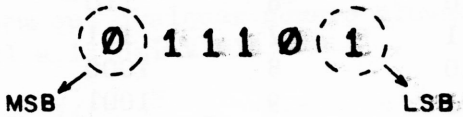
DECIMAL	BINÁRIO	DECIMAL	BINÁRIO
0	0	6	110
1	1	7	111
2	10	8	1000
3	11	9	1001
4	100	10	1010
5	101		

Como já vimos o computador manipula esses números binários através de sinais elétricos H(1) ou L(0). O número decimal 9, binário 1001, seria implementado no computador como visto na figura 1-3.

No registro de números binários o bit na extrema esquerda é denominado MSB ("most significant bit", bit mais significante); na extrema direita temos o LSB ("least significant bit", bit menos significante). A figura 1-4 apresenta uma visualização de MSB e LSB.



1-3 implementação elétrica do binário 1001



1-4 posição de MSB e LSB

Embora muito adequada para máquinas a numeração binária é incômoda para nós humanos. Surgiram assim outros sistemas de numeração que traduzem convenientemente o que está sendo registrado no computador.

Na numeração HEXADECIMAL cada grupo de 4 bits é traduzido para seu valor decimal. Como 4 bits podem atingir até o valor 15 (binário 1111), e para evitar escrever dois números, adotou-se: 10=A, 11=B, 12=D, 13=D, 14=E, 15=F. A tabela abaixo apresenta a equivalência de decimais, binários e hexadecimais.

DECIMAL	BINÁRIO	HEXADECIMAL	DECIMAL	BINÁRIO	HEXADECIMAL
0	0000	0	8	1000	8
1	0001	1	9	1001	9
2	0010	2	10	1010	A
3	0011	3	11	1011	B
4	0100	4	12	1100	C
5	0101	5	13	1101	D
6	0110	6	14	1110	E
7	0111	7	15	1111	F

Mais alguns exemplos são vistos a seguir:

BINÁRIO	HEXADECIMAL	BINÁRIO	HEXADECIMAL
1010 0010	A2	0101 0000	50
1000 1010	8A	0000 1111	OF

Geralmente usa-se a letra H, ou o índice 16, para indicar numeração hexadecimal. Na binária emprega-se a letra B ou o índice 2:

$10100010B = 10100010_2$
$A2H = A2_{16}$

Outro código usado é o BCD ("binary coded decimal", decimal codificado em binário), no qual cada decimal (de 0 a 9) é transformado em um grupo de 4 binários (ou seja, é o processo inverso dos hexadecimais). Exemplos:

DECIMAL	BCD	DECIMAL	BCD
2	0010	15	0001 0101
5	0101	32	0011 0010
10	0001 0000	532	0101 0011 0010

Dizemos NÚMEROS ASSINALADOS quando o MSB não tem valor mas apenas indica o sinal: 1 sinaliza número positivo (+) e 0 número negativo (-). Por exemplo, 1011 é negativo e 0011 é positivo.

Entre os assinalados os números COMPLEMENTO 2 são muito importantes por serem usados nas instruções de microprocessadores. O complemento 2 é número assinalado negativo obtido pela complementação (1 para 0 e vice-versa) do positivo correspondente e somando-se +1. A seguir um exemplo:

POSITIVO: 01001 = +9
COMPLEMENTAÇÃO: 10110
+1: 10111
COMPLEMENTO 2: 10111 = -9

Uma maneira prática de se obter o complemento 2 é tomar o positivo correspondente mantendo o primeiro 1 que aparecer da direita para a esquerda e complementar todos os outros. Exemplo:

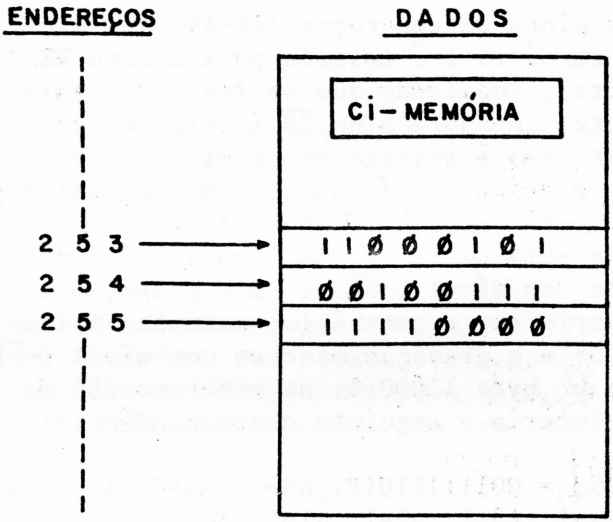
POSITIVO: 01110 = +14
MANTENDO PRIMEIRO 1: ---10
COMPLEMENTANDO OS OUTROS: 10010 = -14

2 - MEMÓRIA

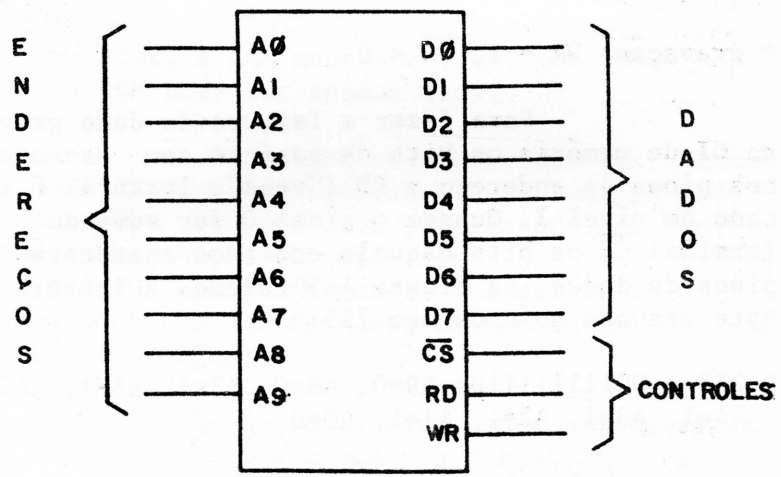
Os bits, significando informações, devem ser armazenados e ficar disponíveis para apresentação quando solicitados. O estágio do computador encarregado desta operação é denominado MEMÓRIA. A memória principal é constituida de CIs, havendo ainda outros meios (fita cassete, diskete) classificados como memória secundária.

Os bits são gravados no CI de memória como sinal lógico 1 ou 0 em posições conhecidas por ENDEREÇOS. Estes partem de zero até a posição mais alta, em ordem crescente. Na figura 1-5 vemos no endereço 253 o byte 11000101; em 254: 00100111; em 255: 11110000. É comum fazer essa apresentação em hexadecimais. O exemplo acima ficaria: endereço FD: C5, FE: 27, FF: F0.

Para gravar dados no CI de memória seus bits são colocados nos respectivos pinos (figura 1-6), a posição onde se quer colocá-los é posta



1-5 endereçamento de dados no CI de memória



1-6 pinagem de CI de memória

nos pinos de endereços (A0-A9). Um sinal de controle em nível 1 é enviado para o pino WR ("write", escrita), indicando que se trata de gravação. Finalmente, quando o pino $\overline{CS}$ ("chip select", seleção de pastilha) é emitido em nível 0 ocorre a gravação. A barra acima do símbolo de um terminal representa que ele está "ativo" quando recebe bit = 0 (e desabilitado para bit = 1). Na figura 1-6 a pastilha é ativada com nível 0-$\overline{CS}$ - e a gravação com nível 1-WR; poderia ser o contrário, pastilha ativada com nível 1-CS- e a gravação ocorrer com nível 0-$\overline{WR}$. A gravação do byte 11000101 no endereço 253 da figura 1-5 obedeceria a seguinte correspondência:

* 253 = 0011111101B; A9=0, A8=0, A7=1, A6=1, A5=1, A4=1, A3=1, A2=1, A1=0, A0=1

* 11000101; D7=1, D6=1, D5=0, D4=0, D3=0, D2=1, D1=0, D0=1

* gravação; WR = 1, $\overline{CS}$ = 0

Para fazer a leitura do dado gravado no CI de memória os bits da posição são assentados nos pinos de endereço e RD ("read", leitura) é colocado em nível 1. Quando o sinal 0 for enviado ao terminal $\overline{CS}$ os bits daquele endereço aparecerão nos pinos de dados. Na figura 1-5 fazemos a leitura do byte gravado no endereço 255;

* 255 = 0011111111B; A9=0, A8=0, A7=1, A6=1, A5=1, A4=1, A3=1, A2=1, A1=1, A0=0

* leitura: RD = 1, $\overline{CS}$ = 0

aparecerá nos pinos de dados

*D7=1, D6=1, D5=1, D4=1, D3=0, D2=1, D1=0, D0=0

Os CIs de memória são classificados em dois tipos: No primeiro - RAM ("random acess memory", memória de acesso aleatório) é possível gravar e fazer leitura dos dados a qualquer momento, facilmente; se numa posição onde está um dado for feita nova gravação os bits antigos serão apagados e no lugar ficarão constando os novos. No segundo tipo - ROM ("read only memory", memória de apenas leitura) os dados são gravados na fábrica, não podendo ser apagados, e o usuário pode fazer apenas leitura dos bits. A PROM ("programable ROM", ROM programável) é uma ROM especial não gravada na fábrica, o que poderá ser feito pelo usuário. Na EPROM ("erasable PROM", PROM apagável) é possível ao usuário tanto gravar como apagar e regravar a ROM. Entretanto, os processos de gravação e apagamento de PROM e EPROM são muito mais lentos e complicados que da RAM, exigindo aparelhos especiais (gravador de EPROM). Para fins práticos, num microcomputador, a RAM pode ser lida, gravada e regravada; a ROM pode ser apenas lida.

As RAMs se subdividem em DINÂMICAS e ESTÁTICAS. Nas primeiras o dado será perdido milésimos de segundos após sua gravação se não for providenciado um circuito que o refresque ("refresh") periodicamente. Nas estáticas o dado permanece gravado enquanto houver tensão de alimentação (voltagem).

A grosso modo, num microcomputador a ROM serve para armazenar o "programa monitor", gravado na fábrica e que orienta toda a atividade do aparelho; na RAM é armazenado o "programa de aplicação", introduzido pelo usuário (através do teclado ou fita cassete) e que usa as sub-rotinas do programa monitor para algum fim determinado.

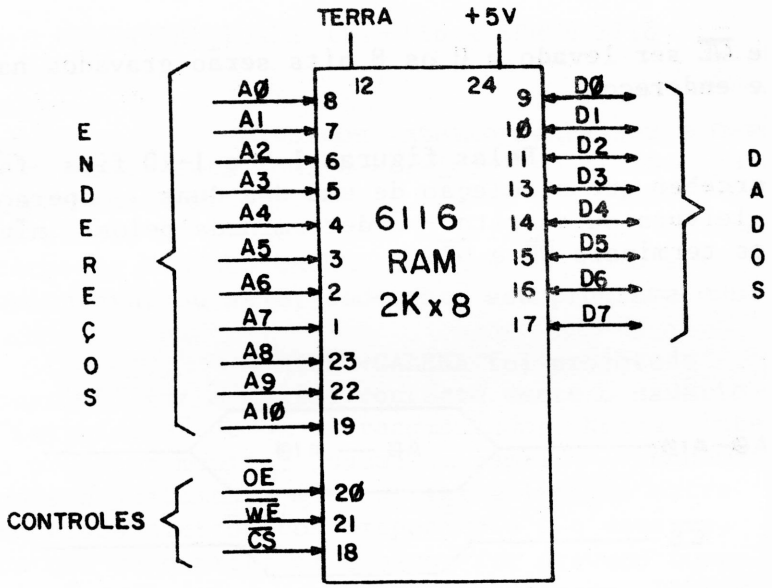
3 - RAM 6116

Na configuração básica do MICRO-GALE NA a função de memória é exercida por um único CI-6116 uma RAM estática 2Kx8. Esta notação indica que ela tem 2K=2.048 endereços e em cada um deles armazena 8 bits = 1 byte.

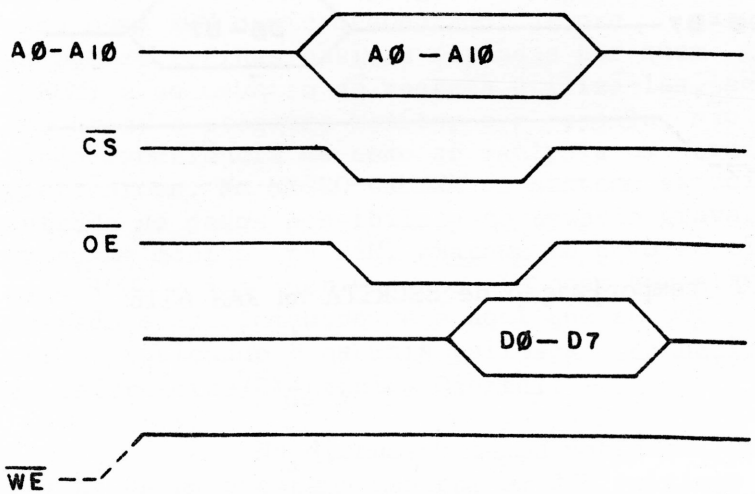
Na figura 1-8 vemos a pinagem da 6116. São necessários 11 pinos de endereços (A0 até A10), sendo o mais baixo 000 0000 0000B = 000 H =0, quando todos pinos de A0 até A10 estão em nível 0; o mais alto é 111 1111 1111B = 7FFH = 2.047, com todos os pinos de A0 até A10 em nível 1. A transferência dos 8 bits de dados é feita através dos pinos D0 até D7, cuja direção (entrando ou saindo da RAM) depende dos terminais de controles $\overline{WE}$ ("write enable", habilita escrita) e $\overline{OE}$ ("output enable", habilita saída). Para leitura dos dados gravados na RAM devemos fazer $\overline{OE}$=0, mantendo $\overline{WE}$=1; para escrita (gravação) na RAM fazemos $\overline{WE}$=0 e mantemos $\overline{OE}$=1. Veja que esses terminais seguem a notação equivalente $\overline{WE}$ = $\overline{WRITE}$ = $\overline{WR}$ e $\overline{OE}$ = $\overline{READ}$ = $\overline{RD}$. Em ambos os casos é necessário colocar previamente $\overline{CS}$ ("chip select", seleção de CI) em nível 0.

A temporização de leitura na 6116 é vista na figura 1-9. O endereço onde se deseja fazer uma leitura é assentado nos terminais A0-A10, $\overline{CS}$ é mantido em 0 e $\overline{WE}$ em 1; pouco tempo depois de $\overline{OE}$ ser levado a 0 os 8 bits gravados naquela posição aparecerão nos terminais D0-D7.

Na escrita (figura 1-10) o endereço onde se gravará é assentado nos terminais A0-A10, o byte a ser gravado é colocado nos terminais D0-D7, $\overline{CS}$ é mantido em 0 e $\overline{OE}$ em 1; pouco tempo depois



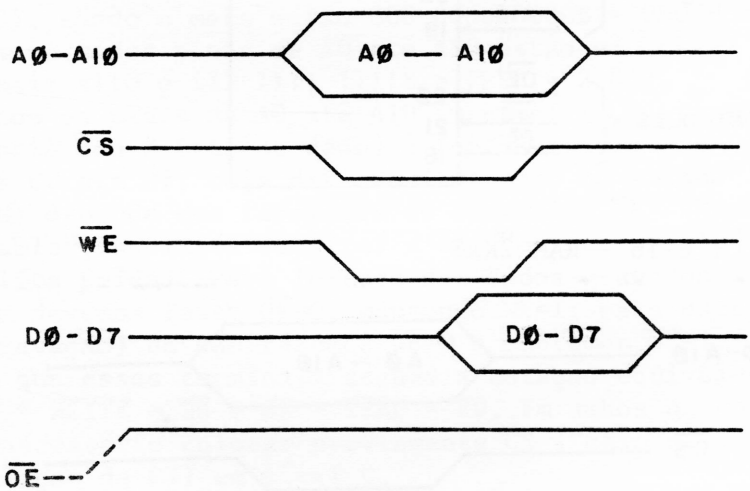
1-8 CI 6116 - RAM 2Kx8



1-9 temporização de LEITURA na RAM 6116

de $\overline{WE}$ ser levado a 0 os 8 bits serão gravados naque_ le endereço.

Pelas figuras 1-9 e 1-10 fica fácil perceber que a seleção de uma das duas operações - leitura ou escrita - é determinada pelos níveis dos terminais $\overline{WE}$ e $\overline{OE}$.



1-10 temporização de ESCRITA na RAM 6116

4 - COMPATIBILIDADE RAM 6116 x EPROM 2716

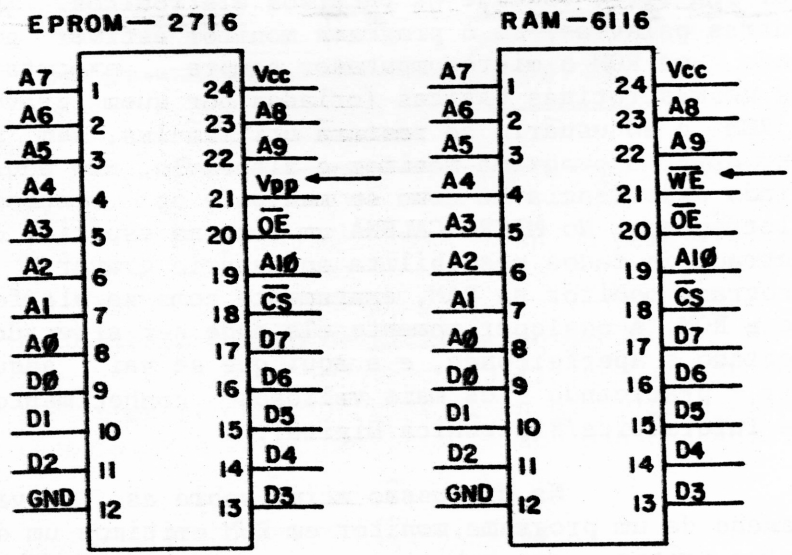
Dissemos anteriormente ser a memória ROM que orienta toda a atividade do microcomputador (programa monitor); vimos também que o MICRO-GALENA só dispõe de memória RAM. O leitor então se perguntará: se o MICRO-GALENA não usa ROM como ele se orientará, ou seja, onde está seu programa monitor.

O MICRO-GALENA foi projetado para possibilitar o máximo contacto entre o usuário e a "intimidade" de um microcomputador. Nessa intenção o programa monitor assume um papel fundamental, pois é justamente ele que faz a ligação entre os recursos eletrônicos disponíveis e o usuário. Ocorre que se o programa monitor for gravado numa ROM ele é definitivo (ou de difícil e cara alteração, como na EPROM), não permitindo ao usuário testar por sua livre vontade os recursos eletrônicos. Em outras palavras, se o programa monitor estiver gravado numa ROM o microcomputador sempre executará as mesmas rotinas básicas (criadas por quem gravou a ROM) e ao usuário só restará utilizá-las, sem poder criar o programa monitor e alterá-lo, não adquirindo experiência de como se manipula os recursos eletrônicos. No MICRO-GALENA um sistema especial de entrada de dados possibilita ao usuário gravar o programa monitor na RAM, exatamente como se ela fosse a ROM. A qualquer momento ele pode ser alterado, testado e aperfeiçoado, e é aqui que se vai adquirir - praticando - os mais valiosos conhecimentos de Informática/Eletrônica Digital.

Na discussão acima sobre as desvantagens de um programa monitor em ROM omitimos um detalhe prático, que habitualmente torna-se uma des-

vantagem definitiva: a gravação da ROM exige equipamentos especializados, geralmente caros e complexos. Em São Paulo existem empresas que fazem a gravação, o que facilita bastante, mas em outros locais o interessado teria de comprar o gravador/copiador de EPROM.

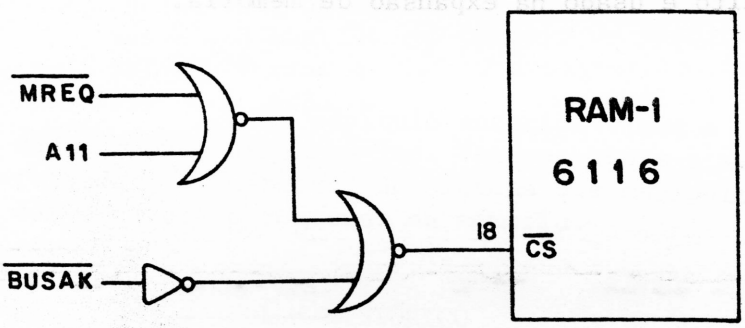
Mas o MICRO-GALENA também foi projetado para máxima versatilidade, incluindo aplicações profissionais. Deste ponto de vista a ROM é obrigatória, já que numa aplicação profissional não é conveniente ao usuário dar entrada nos dados do programa monitor toda vez que precisar usá-lo. A solução implementada no MICRO-GALENA é utilizar uma RAM (6116) totalmente compatível com uma EPROM (2716). Durante a aprendizagem coloca-se a RAM, nas aplicações profissionais a EPROM. O próprio MICRO-GALENA pode ser empregado como gravador de EPROM.



1-11 compatibilidade EPROM 2716 x RAM 6116

5 - CIRCUITO DE HABILITAÇÃO DA MEMÓRIA

O circuito de habilitação da memória RAM é visto na figura 1-12, consistindo num sistema com três entradas ($\overline{MREQ}$, A11, $\overline{BUSAK}$) que permitem dois modos de habilitação.



1-12 circuito de habilitação da memória

No primeiro modo $\overline{BUSAK}$ é levado a zero; ao ser invertido e aparecer como 1 na 2a. NOR resulta nível zero na saída desta, habilitando a RAM independente dos outros dois terminais de entrada.

No 2º modo $\overline{MREQ}$ e A11 são simultaneamente levados a zero, ocasionando 1 na saída da 1a. NOR e em conseqüência zero na saída da 2a. NOR inde

pendente da entrada $\overline{BUSAK}$. Veja que se somente uma das entradas do par A11-$\overline{MREQ}$ for levada a zero não ocorrerá a habilitação.

Neste esquema a RAM-1 fica no endereço 0XXX XXXX XXXX, onde o zero no MSB corresponde a A11 e os 'x' aos bits de A0 até A10. Portanto o menor endereço da RAM-1 será 0000 0000 0000 B=0 e o maior 0111 1111 1111 B=2047, sempre com A11=0.

Mais adiante veremos que parte deste circuito é usado na expansão de memória.

CAPÍTULO 2
MICROPROCESSADOR

No capítulo anterior vimos o que são dados e onde armazená-los. Veremos agora o microprocessador, o "cérebro" do sistema que manipula os dados e altera o conteúdo da memória.

1 - HISTÓRICO

O microprocessador foi inventado em 1969 e a partir daí desenvolveu-se rapidamente, assumindo posição dominante em Eletrônica Digital. Hoje ele é usado como "cérebro" de microcomputadores, vídeo-games, telefones, instrumentos, robôs, televisores, automóveis e outros.

O primeiro microprocessador foi o 4004 da INTEL, classificado como "de 4 bits" por manipular 4 bits de dados por vez. O 8080 - também da INTEL e de 8 bits - o substituiu logo em seguida, sendo ainda largamente empregado. Encontramos nesta geração além de 8080 o 6502 - MOSTEK, famoso por ser empregado nos micros APPLE, e o 6800 - MOTOROLA. Entretanto a grande vedete dos microprocessa-

dores de 8 bits é o Z-80 da ZILOG, lançado em 1976, utilizado nos micros TK, CP, e numa infinidade de máquinas e equipamentos.

Atualmente já estão em atividade os microprocessadores de 16 bits que manipulam 16 bits de cada vez. São muito mais caros e complexos, sendo encontrados somente em máquinas sofisticadas. Nesta categoria destacam-se o 8088 da INTEL (usado no IBM-PC) e o 68.000 da MOTOROLA (adotado pelo McINTOSH da APPLE); este último é apontado como o mais potente de sua geração.

2 - O MICROPROCESSADOR EM AÇÃO

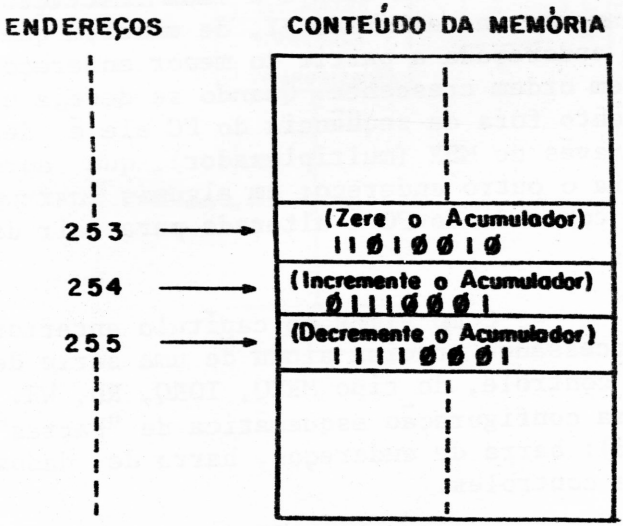
Chamamos "instrução" um conjunto codificado de bits, de 1 a 4 bytes (portanto 8 a 32 bits). O microprocessador é capaz de executar a operação lógica - aritmética indicada pelo código da instrução, sendo que cada marca tem sua própria lista de instruções - cerca de 100. Organizando-se adequadamente a seqüência de instruções obtem-se um "programa", isto é, uma série de operações lógico-aritméticas executadas pelo microprocessador. Programar consiste exatamente em fazer uma organização de seqüência que leve a um resultado desejado.

Os bits das instruções (obtidos no manual do fabricante do microprocessador) são gravados como níveis 1 ou 0 na memória; na ordem em que devem ser executados. O microprocessador "lê" a 1a. instrução e executa-a, em seguida a 2a., 3a., etc, até executar toda a seqüência que é o programa.

A figura 2-1 ilustra um programa hipotético gravado na memória. No endereço 253 deci-

mal é escrito o código 11010010 que o microprocessador identificará como instrução "zere o acumulador"; ele então deixa seu acumulador com valor zero. Em seguida ele lê a instrução do endereço 254 - "incremente o acumulador", e ao executá-la resulta acumulador=0+1=1. No próximo endereço (255) o microprocessador encontrará "decremente o acumulador" e fará acumulador=1-1=0. Observe que anotamos os endereços em numeração decimal, mas no circuito o endereçamento será feito em binários, como vimos no capítulo anterior.

Se a memória for uma ROM o microprocessador poderá apenas ler as instruções gravadas. Mas se for RAM ele poderá ler, alterar e eventulamente deixar o resultado das operações armazenado num endereço.



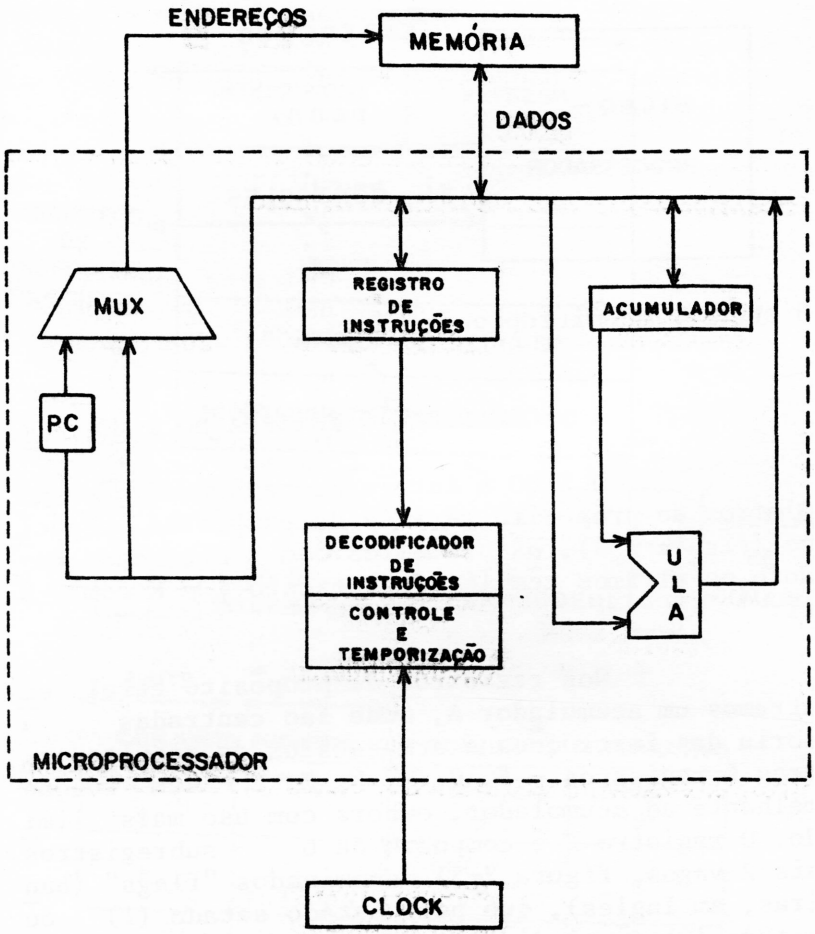
2-1 programa na memória

3 - ARQUITETURA DE MICROPROCESADOR

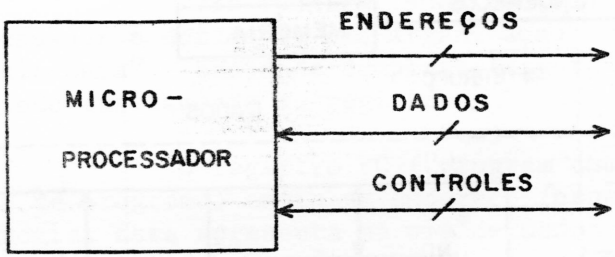
Para realizar suas funções o microprocessador é dotado de uma organização interna- "arquitetura" - bastante complexa (figura 2-2), consistindo basicamente de registros.

O registro PC ("program counter",contador de programa) envia um endereço (binário) para a memória. Esta apresenta na via de dados os bits da instrução referente àquele endereço, que são copiados num registro de instruções e decodificados. Os resultados são obtidos através de operações lógicas entre os dados existentes no acumulador e outros que estejam na via de dados, sendo realizadas na ULA (unidade lógica-aritmética);geralmente a resposta é armazenada no acumulador. As instruções são executadas passo-a-passo segundo a sincronização de um sinal externo denominado "clock" (relógio). O PC é inicializado com zero e a cada instrução é automaticamente incrementado +1, de maneira que a memória é endereçada a partir do menor endereço até o maior, em ordem crescente. Quando se deseja um endereçamento fora da seqüência do PC ele é desabilitado através do MUX (multiplexador), que apresenta à memória o outro endereço; em algumas instruções o próprio conteúdo do PC é alterado para sair da seqüência.

Como vimos no capítulo anterior o microprocessador precisa ainda de uma série de sinais de controle, do tipo MREQ, IORQ, RD, WR. Temos assim uma configuração esquemática de "barras" (figura 2-3): barra de endereços, barra de dados e barra de controles.



2-2 arquitetura de microprocessador



2-3 barras do microprocessador

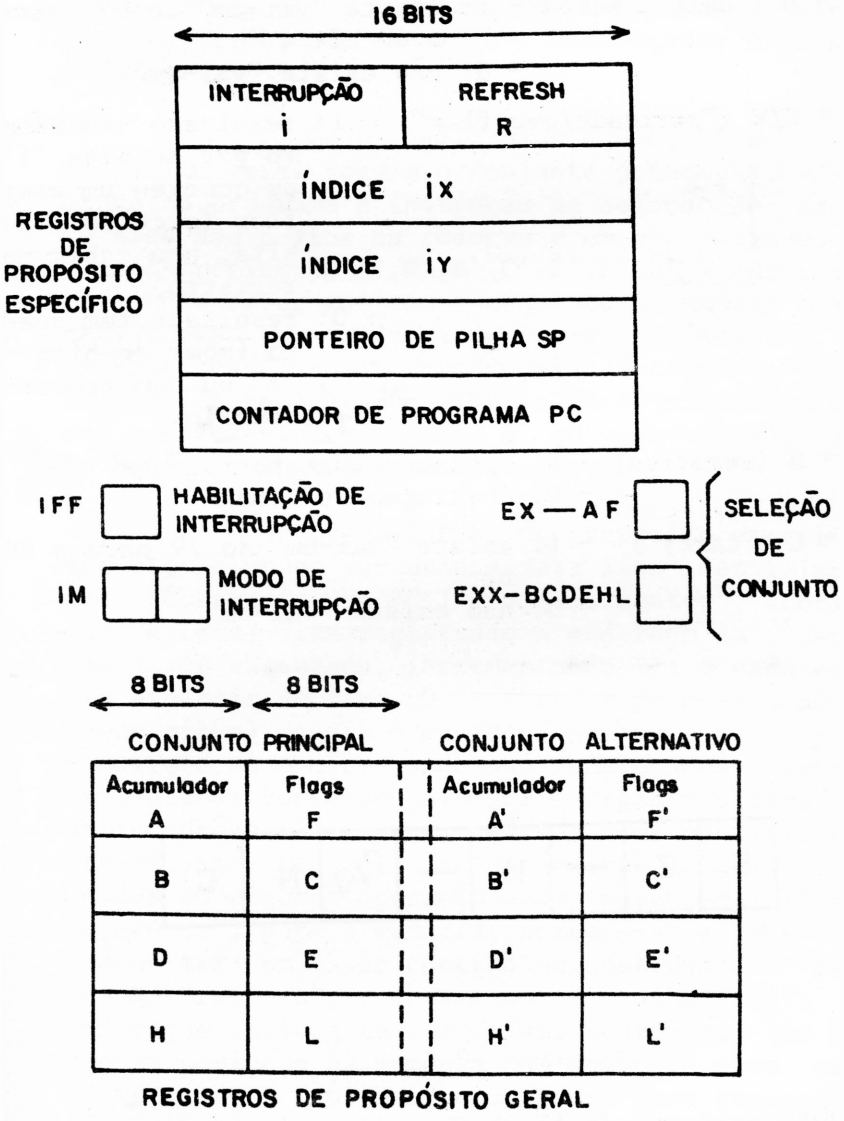
4 - ORGANIZAÇÃO INTERNA DO Z-80

O Z-80 é internamente organizado em registros de propósito geral e de propósito específico (figura 2-4), os primeiros com 8 bits cada enquanto os últimos têm 16 bits, exceto I e R com 8 bits.

Nos registros de propósito geral encontramos um acumulador A, onde são centradas a maioria das instruções e armazenados os resultados. Outros 6 registros - B, C, D, E, H, L - têm função semelhante ao acumulador, embora com uso mais limitado. O registro F é composto de 6 subregistros (mais 2 vagos, figura 2-5) denominados "flags" (bandeiras, em Inglês), que pelo estado setado (1) ou resetado (0) dão indicação sobre o resultado da última instrução executada:

* S (sinal) - 1: resultado negativo
  - 0: resultado positivo

* Z (zero) - 1: conteúdo de registro igual a byte comparado



2-4 registros do Z-80

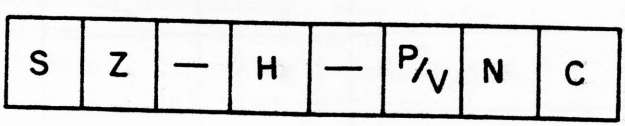
-0: conteúdo e byte desiguais

* H ("half", meio) - 1: existe "vai-um" do 3º para o 4º bit
  - 0: não existe "vai-um"

* P/V ("paridade/overflow") - 1: resultado tem número par de bits '1' ou ocorreu um overflow (excesso de bits, não cabem no registro)
  - 0: resultado tem número ímpar de bits "1" ou não ocorreu overflow

* N (negativo) - 1: instrução de subtração
  - 0: instrução de soma

* C ("carry") - 1: existe "vai-um" do 7º para o 8º bit
  - 0: não existe "vai-um"



2-5 registro de flags F

Os registros (exceto A e F) podem formar pares de 16 bits: BC, DE e HL. Este último é muito usado como ponteiro ("endereçador") para a memória.

Os 8 registros A,F,B,C,D,E,H,L de propósito geral formam o conjunto principal. Colocando no programa a instrução de seleção de conjunto eles são postos na reserva e em seus lugares entram os registros A',F',B',C',D',E',H',L' do conjunto alternativo, que tem exatamente as mesmas características. Essa mudança de conjuntos facilita ao microprocessador ser interrompido durante um programa. Na figura 2-4 os registros EX-AF e EX-BCDEHL são complementados (1 para 0, ou vice-versa), quando ocorre a instrução de seleção.

Os registros de propósito específico são 6 e só podem ser usados para fins determinados pelo fabricante. No PC ("program counter", contador de programa) fica registrado o endereço da instrução a ser executada, incrementando + 1 a cada execução a partir de zero. O SP ("stack pointer", ponteiro de pilha) guarda o endereço presente da pilha - posição na memória onde são armazenados temporariamente os conteúdos de outros registros para que o microprocessador possa usá-los numa operação de interrupção. Os registros de índices IX e IY são usados em algumas operações como ponteiros ("endereçadores") para a memória, somando-se a seus conteúdos mais um valor contido na instrução. O registro R ("refresh", refrescamento) contém 7 bits de endereços válidos no fim de uma instrução que selecionam posições na memória RAM dinâmica para serem "refrescados" (voltagem e corrente para renovar o nível lógico 1 ou 0), caso contrário os bits gravados se perdem em questões de milisegundos (memó-

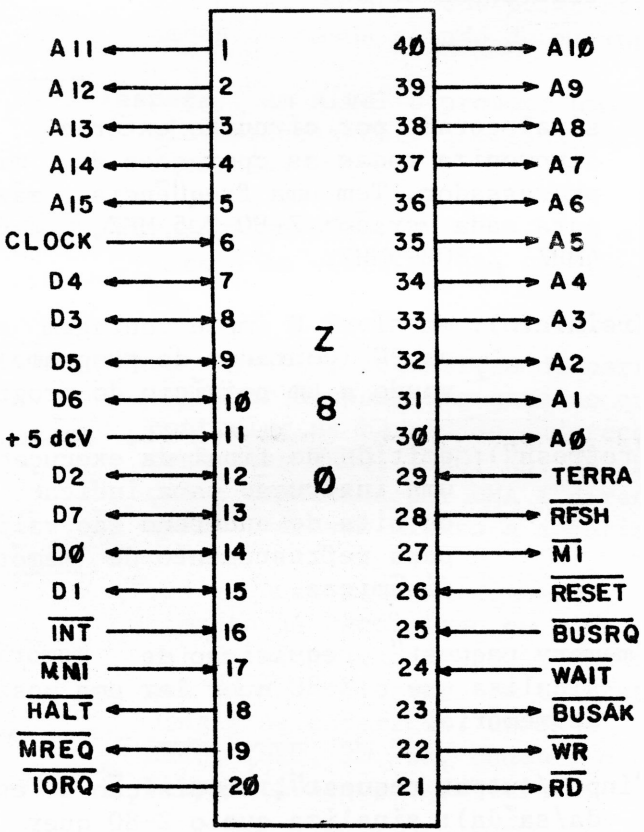
rias que usem RAM estática não precisam de "refresh"). Uma instrução no programa habilita interrupção, isto é, permite que um dispositivo externo interrompa o processamento de um programa pelo microprocessador, que passa a atendê-lo; neste caso o registro IFF é setado. Outra instrução desabilita interrupções, ressetando IFF. Há três modos (IM0, IM1, IM2) de interrupção, com operações diferentes, selecionadas por instruções e que setam ou resetam correspondentemente o registro IM de dois bits (00, 01 ou 10). Se IFF está setado e IM no modo 2 o dispositivo externo coloca 1 byte na barra de dados, constituindo o byte inferior (A0-A7) que é juntado ao byte superior (A8-A15) existente no registro I, formando o endereço de 16 bits onde se encontra a rotina de interrupção a ser executada.

5 - PINAGEM DO Z-80

O Z-80 é um CI de 40 pinos, como ilustrado na figura 2-6 (na verdade temos 3 versões Z-80, Z80A, Z-80B - que diferem só na velocidade de execução das instruções). Lembre-se que terminais com uma barra no símbolo estão ativos com nível 0 e desativados com nível 1.

* A0-A15: 16 terminais que formam um endereço binário de 0 a 65.536 (0000 0000 0000 0000 B a 1111 1111 1111 1111B, ou 0000 H a FFFF H).
São conectados à barra de endereços e neles os sinais sempre têm o sentido do microprocessador para o resto do sistema.

* D0-D7: 8 terminais de dados, com sentido do microprocessador para o resto do sistema



2-6 pinagem do Z-80

(escrita na memória ou saída em porta)ou do resto do sistema para o microprocessador (leitura de memória ou entrada por porta). São conectados à barra de dados.

* +5Volts: voltagem de alimentação, dc regulada com tolerância de 5%.

* TERRA

* CLOCK: sinal gerado por circuito externo que sincroniza todas as operações do microprocessador. Tem uma freqüência máxima para cada versão: Z-80-2,5 MHZ, Z-80A-4MHZ, Z-80B-5MHZ.

* $\overline{\text{RESET}}$ (reinício): em nível 0 impõe contagem zero no PC (contador de programa), levando a um reinício do programa.

* $\overline{\text{RFSH}}$ ("refresh"): emitido no final da execução de uma instrução para indicar que os bits de endereço são válidos para refrescamento de memórias dinâmicas.

* $\overline{\text{MREQ}}$ ("memory request", requisição de memória): sinaliza que o Z-80 quer ler uma posição de memória.

* $\overline{\text{IORQ}}$ ("input/output request", requisição de entrada/saída): sinaliza que o Z-80 quer dar entrada ou saída num byte de dados através de uma porta de entrada ou saída, proveniente ou com destino a um dispositivo externo.

* $\overline{RD}$ ("read", leitura): leitura de dados da memória ou entrada de dados pelas portas, em ambos os casos em direção ao microprocessador.

* $\overline{WR}$ ("write", escrita): escrita de dados na memória ou saída de dados pelas portas, em ambos os casos saindo do microprocessador

* $\overline{WAIT}$ (espera): em nível 0 provoca uma espera no microprocessador, que fica parado sem executar instruções

* $\overline{HALT}$ (parada): em nível 0 indica que o Z-80 está parado em vista de uma instrução de parada

*$\overline{BUSRQ}$ ("bus request", requisição de barra): é emitido por dispositivo externo que quer controlar as barras de endereços, dados e controles $\overline{MREQ}$, $\overline{IORQ}$, $\overline{RD}$, $\overline{WR}$. Esta operação é conhecida por DMA ("direct memory acess", acesso direto à memória)

* $\overline{BUSAK}$ ("bus request acknowledgement", reconhecimento de requisição de barras): em nível 0 indica que o Z-80 reconheceu o sinal $\overline{BUSRQ}$, colocando em tristate seus terminais de endereços, dados e controles $\overline{MREQ}$, $\overline{IORQ}$, $\overline{RD}$, $\overline{WR}$, cedendo o controle ao dispositivo que o solicitou.

* $\overline{INT}$ (interrupção): é emitido por dispositivo externo que deseja interromper a execução do programa pelo microprocessador e fazê-lo executar uma outra seqüência de instruções. O sinal $\overline{INT}$ só é reconhecido pelo

Z-80 se o registro IFF estiver setado por uma instrução do programa.

* $\overline{NMI}$ ("no maskable interrupt", interrupção não mascarável): semelhante a $\overline{INT}$, porém é reconhecido independente do estado de IFF. Geralmente é usado para execução de rotinas de emergência (do tipo falta de energia).

* $\overline{MI}$ ("machine 1"): sinaliza que o Z-80 acaba de executar um ciclo de busca ("fetch") de instrução na memória.

6 - TEMPORIZAÇÃO DO Z-80

Todas as operações do Z-80 são realizadas num intervalo de tempo dependente da freqüência do sinal de clock percebido no pino 6. Para o Z-80A a freqüência especificada é 4 MHZ, de onde resulta o período de clock T (tempo gasto em 1 ciclo do clock, figura 2-7):

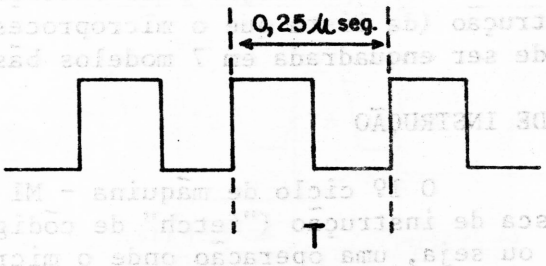
$$\text{período (micro-segundos)} = \frac{1}{\text{freqüência (em Mega-Herz)}}$$

$$T = \frac{1}{4} \text{ --- } T = 0{,}25 \text{ micro-segundos}$$

(onde micro-segundos é 1 segundo dividido por 1 milhão, e 1 Mega-Herz é 1 Herz vezes 1 milhão).

Note que se usarmos um clock com freqüência abaixo desta o período de clock T aumentará. Por exemplo, para 1 MHZ:

$$T = \frac{1}{1} \text{ --- } T = 1 \text{ micro-segundos}$$

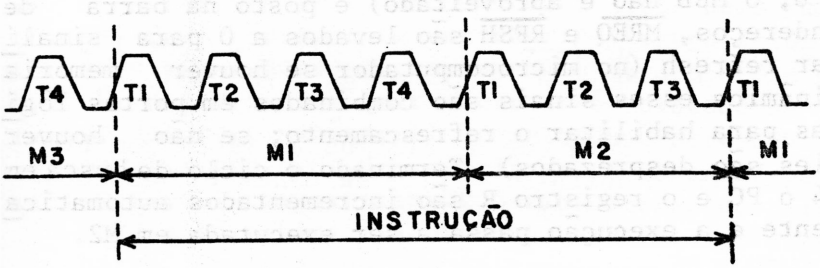


2-7 período de clock T para freqüência 4 MHZ.

Uma única instrução demora vários períodos de clock, que são numerados T1, T2, T3, etc (figura 2-8). A execução também é feita em etapas denominadas ciclos de máquina M1, M2, etc. No exemplo da figura 2-8 a instrução é executada em 2 ciclos de máquina (M1 e M2), compostos de 4 e 3 períodos de clock (T1, T2, T3, T4/T1, T2, T3), respectivamente. Se estivéssemos usando o Z-80A de 4 MHZ esta instrução seria executada no tempo de

(4+3) períodos de clock x 0,25 microsegundos = 1,75 microsegundos

(observe que o microprocessador executa cerca de 600.000 instruções como essa em 1 segundo!).

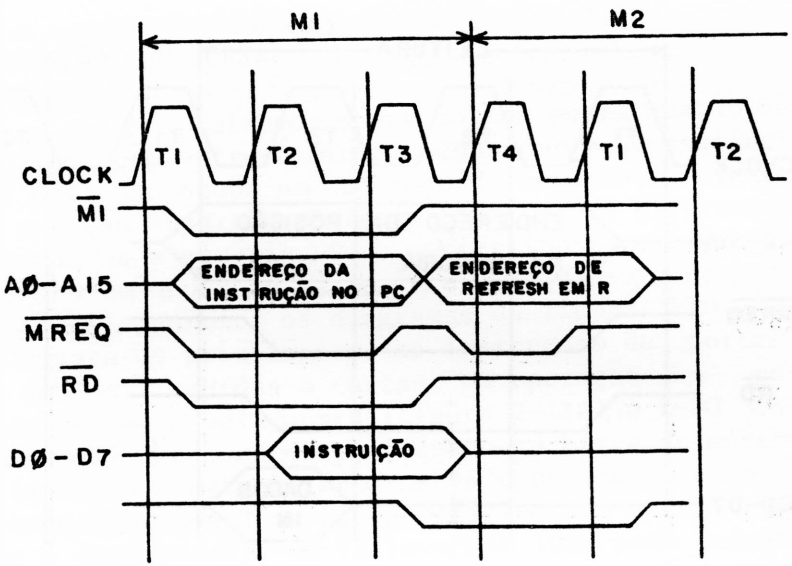


2-8 ciclo de instrução, ciclo de máquina (M) e período de clock (T)

A temporização para execução de qualquer instrução (da lista que o microprocessador obedece) pode ser enquadrada em 7 modelos básicos:

* BUSCA DE INSTRUÇÃO

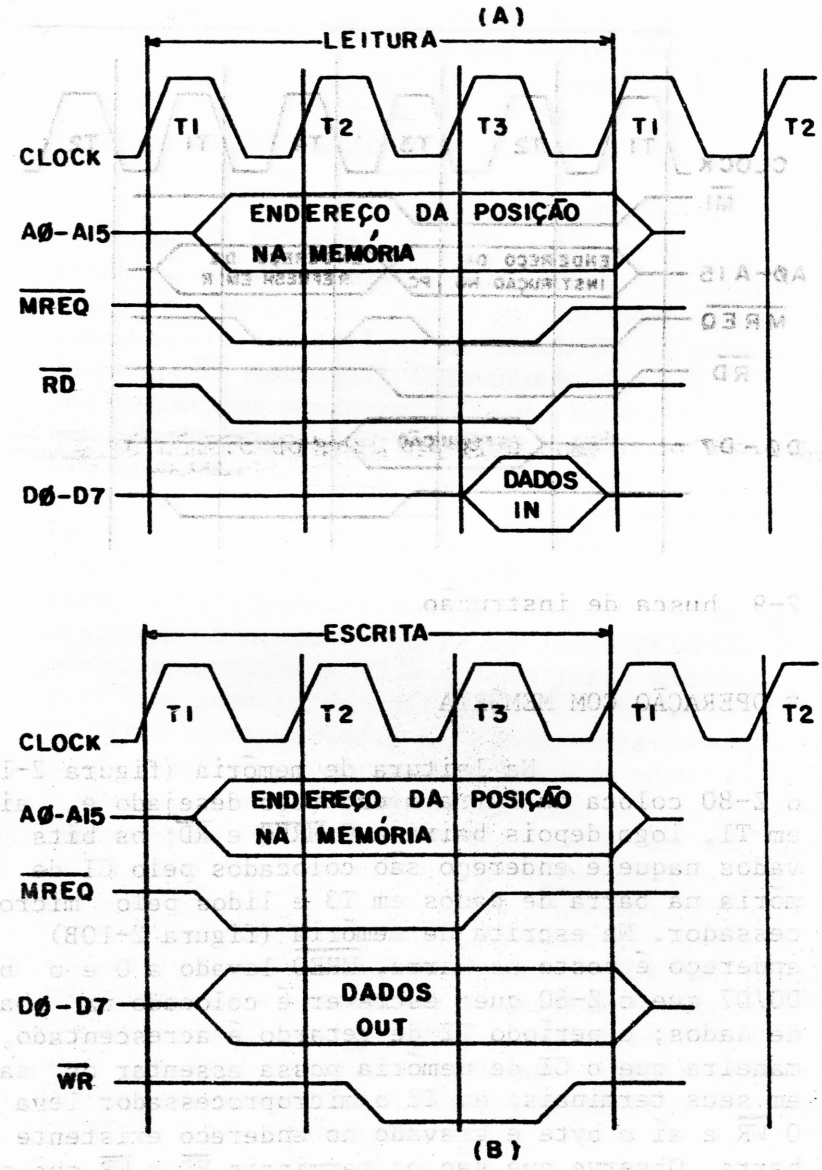
O 1º ciclo de máquina - M1 - sempre é uma busca de instrução ("fetch" de código operacional), ou seja, uma operação onde o microprocessador lê na memória o código da instrução que deve executar. No 1º período de clock T1 (figura 2-9) o terminal $\overline{M1}$ (pino 27) é levado a nível 0 pelo Z-80, indicando que se trata do ciclo de busca de instrução. Neste mesmo instante o conteúdo do PC (registro contador de programa) é colocado na barra de endereços, correspondendo à posição na memória onde está gravado o código da instrução. Pouco depois, ainda em T1, os terminais $\overline{MREQ}$ e $\overline{RD}$ são levados a zero, sinalizando leitura de memória. O código da instrução é posto na barra de dados pelo CI de memória no começo de T3 e é lido pelo microprocessador, que passa a decodificá-lo ("decifrá-lo") nos próximos dois períodos de clock. Durante este tempo de decodificação as barras ficariam ociosas. O Z-80 aproveita então para realizar o refrescamento de memórias dinâmicas: o conteúdo do registro de refrescamento R (os 7 bits menos significativos 0 a 6, o MSB não é aproveitado) é posto na barra de endereços, $\overline{MREQ}$ e $\overline{RFSH}$ são levados a 0 para sinalizar refresh (no microcomputador se houver memória dinâmica esses sinais são combinados em portas lógicas para habilitar o refrescamento; se não houver eles são desprezados). Terminado o ciclo de busca em T4 o PC e o registro R são incrementados automaticamente e a execução passa a ser executada em M2.



2-9 busca de instrução

* OPERAÇÃO COM MEMÓRIA

Na leitura de memória (figura 2-10A) o Z-80 coloca na barra o endereço desejado e, ainda em T1, logo depois baixa a 0 $\overline{\text{MREQ}}$ e $\overline{\text{RD}}$; os bits gravados naquele endereço são colocados pelo CI de memória na barra de dados em T3 e lidos pelo microprocessador. Na escrita de memória (figura 2-10B) o endereço é posto na barra, $\overline{\text{MREQ}}$ levado a 0 e o byte D0/D7 que o Z-80 quer escrever é colocado na barra de dados; o período T2 de retardo é acrescentado, de maneira que o CI de memória possa assentar os dados em seus terminais; em T2 o microprocessador leva a 0 $\overline{\text{WR}}$ e aí o byte é gravado no endereço existente na barra. Observe que são os terminais $\overline{\text{RD}}$ e $\overline{\text{WR}}$ que selecionam entre escrita ou leitura.

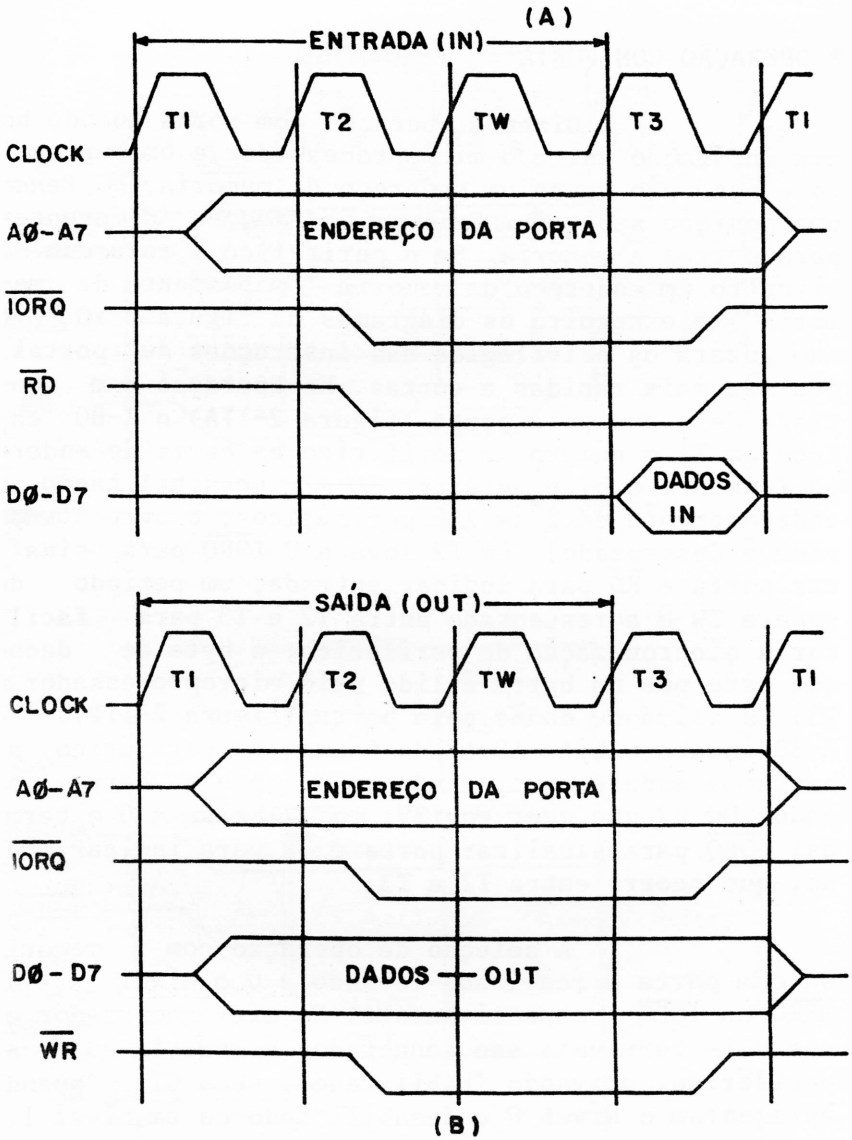


2-10 operação com memória (A) leitura (B) escrita

* OPERAÇÃO COM PORTA

Dizemos operação com porta quando há uma interação entre o microprocessador e um periférico e este não ocupa um endereço de memória, tendo uma posição separada. Existem instruções diferentes para portas e memória. Se o periférico é colocado no circuito em endereço de memória ("mapeamento de memória") ele seguirá os diagramas da figura 2-10 e não gozará os privilégios das instruções de portas, que são mais rápidas e curtas. Na operação com entrada de dados pela porta (figura 2-11A) o Z-80 coloca em T1 o número do periférico na barra de endereço (somente o byte inferior A0-A7, possibilitando o endereçamento de 2^8 = 256 periféricos; o byte superior é desprezado), em T2 leva a 0 $\overline{IORQ}$ para sinalizar porta e $\overline{RD}$ para indicar entrada; um período de espera TW é acrescentado entre T2 e T3 para facilitar a sincronização do periférico; o byte de dados que este põe na barra é lido pelo microprocessador em T3. Na saída de dados pela porta (figura 2-11B) o Z-80 coloca o byte A0-A7 do número do periférico na barra de endereços e, ainda em T1, põe na barra os dados D0-D7 que quer enviar; em T2 baixa a 0 o terminal $\overline{IORQ}$ para sinalizar porta e $\overline{WR}$ para indicar saída, que ocorre entre T2 e T3.

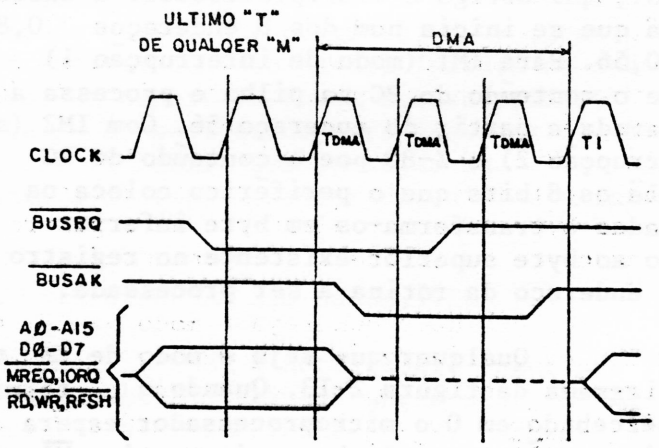
A seleção de operação com memória ou com porta é realizada levando a 0 o nível de $\overline{MREQ}$ ou $\overline{IORQ}$ respectivamente. No microcomputador esses dois terminais são conectados à memória ou ao periférico, ativando (habilitando) seus CIs quando apresentam o nível 0 e desabilitando-os em nível 1.



2-11 operação com porta (A) entrada (B) saída

* DMA

No DMA ("direct memory acess", acesso direto à memória; figura 2-12) um periférico ativa o pino 23 do microprocessador - $\overline{BUSRQ}$ ("bus request", requisição de barras), requisitando para si o controle das barras. O Z-80 termina o ciclo de máquina M que estava executando e no próximo período de clock T ativa o terminal $\overline{BUSAK}$ ("bus request acknowledgement", reconhecimento da requisição de barras), indicando que reconheceu o pedido feito pelo periférico e está passando a ele as barras; a partir deste momento ficam em tristate os pinos A0-A15, D0-D7, $\overline{MREQ}$, $\overline{IORQ}$, $\overline{RD}$, $\overline{WR}$, $\overline{RFSH}$. O periférico passa a controlar as barras, lendo ou escrevendo diretamente na memória sem a interferência do microprocessador, que fica ocioso. A operação DMA pode se estender por quanto tempo for preciso (na figura 2-12 registramos apenas 3 períodos de clock para DMA). Quando quiser terminá-la o periférico desativa $\overline{BUSRQ}$; no clock seguinte o Z-80 desativa $\overline{BUSAK}$ e no próximo volta a controlar as barras.



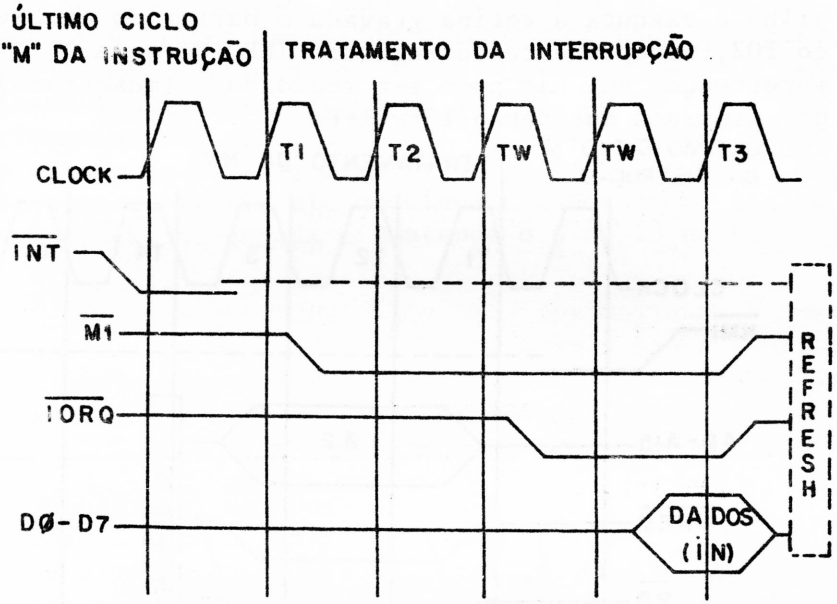
2-12 DMA - acesso direto à memória

* INTERRUPÇÃO

Na figura 2-4 vimos os registros internos IFF e IM do Z-80. Uma instrução no programa permite setar ou resetar IFF. Com IFF em reset (0) qualquer sinal de interrupção vindo de um periférico que cheque ao pino 16 - $\overline{INT}$ - do microprocessador é desconhecido. Mas se estiver setado (1) o programa é interrompido temporariamente e tem início uma rotina de interrupção cujas instruções estão gravadas em algum endereço da memória. Este endereço é programado por uma das instruções IM0, IM1, IM2.

Programando IM0 (modo de interrupção 0) quando o periférico envia um sinal 0 para o terminal $\overline{INT}$ do Z-80 ele interrompe o processamento do programa no fim da instrução presente, coloca o conteúdo do PC no endereço indicado pelo SP (ponteiro de pilha) e executa a instrução que for posta na barra de dados. Esta geralmente é RST ("restart", reinício), que obriga o microprocessador a executar a rotina que se inicia num dos 8 endereços 0,8,16, 24,32,40,56. Para IM1 (modo de interrupção 1) o Z-80 põe o conteúdo do PC na pilha e processa a rotina gravada a partir do endereço 56. Com IM2 (modo de interrupção 2) o Z-80 põe o conteúdo do PC na pilha, lê os 8 bits que o periférico coloca na barra de dados e transforma-os em byte inferior que agrupado ao byte superior existente no registro I, forma o endereço da rotina a ser processada.

Qualquer que seja o modo de IM teremos o diagrama da figura 2-13. Quando o terminal $\overline{INT}$ é percebido em 0 o microprocessador espera o fim da instrução e no clock seguinte ativa $\overline{M1}$; dois clocks após $\overline{IORQ}$ também é ativado. Compete ao peri-



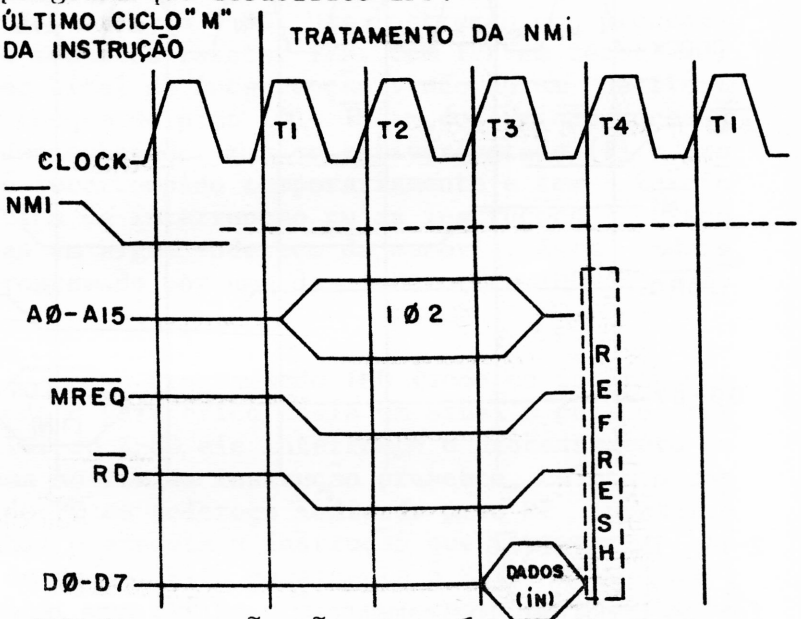
2-13 INTERRUPÇÃO

férico combinar logicamente $\overline{M1}$ e $\overline{IORQ}$ para habilitar o envio de dados, que ocorrerá no 4º clock. Observe que se estiver IM1 não haverá leitura de dados. Dois períodos de espera TW foram acrescentados, dando mais tempo ao periférico para assentar os sinais. A partir de T3 ocorre uma operação de refrescamento enquanto o microprocessador decodifica a instrução.

* NMI - INTERRUPÇÃO NÃO MASCARÁVEL

NMI ("no maskable interrupt") é uma operação especial de interrupção destinada a casos de emergência. É semelhante a IM1, apenas alterando o endereço da rotina de interrupção. Quando o pino 17 $\overline{NMI}$ é ativado o Z-80 coloca o conteúdo do PC na

pilha e executa a rotina gravada a partir do endereço 102, independente do registro IFF. Ou seja, a interrupção NMI não pode ser impedida (mascarada) por programa que desabilite IFF.



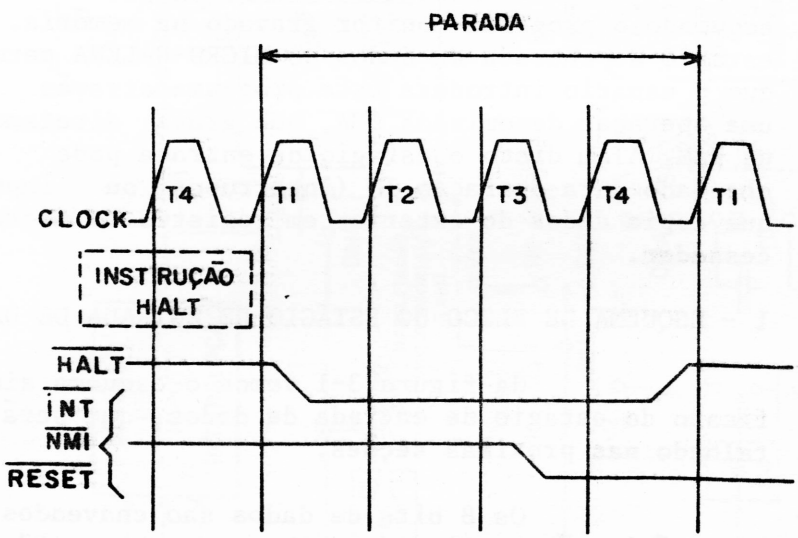
2-14 NMI interrupção não mascarável

O diagrama da NMI é apresentado na figura 2-14. Quando o terminal NMI é percebido em 0 o microprocessador termina a instrução que estava executando, coloca 102 na barra de endereço, ativa $\overline{MREQ}$ e $\overline{RD}$ sinalizando leitura de memória naquela posição e dois clock após ler o 1º byte do código da instrução que é posto na barra de dados. Enquanto a decodificação é realizada ocorre uma operação de refresh.

* OPERAÇÃO DE PARADA- HALT

Quando processa a instrução HALT o

Z-80 pára o programa, apenas realizando o refresh. Após os 4 clocks dedicados à instrução HALT (figura 2-15) o microprocessador ativa o pino 18-$\overline{\text{HALT}}$- e permanece 4 clocks em estado de parada. Se em T3 for ativado um dos pinos $\overline{\text{INT}}$, $\overline{\text{NMI}}$ ou $\overline{\text{RESET}}$ o Z-80 sái do estado de parada, desativando o terminal $\overline{\text{HALT}}$ em T4; caso contrário ele continuará executando ciclos de 4 clocks de parada e mantendo o pino 18 em 0.



2-15 HALT (parada)

CAPÍTULO 3
ENTRADA DE DADOS

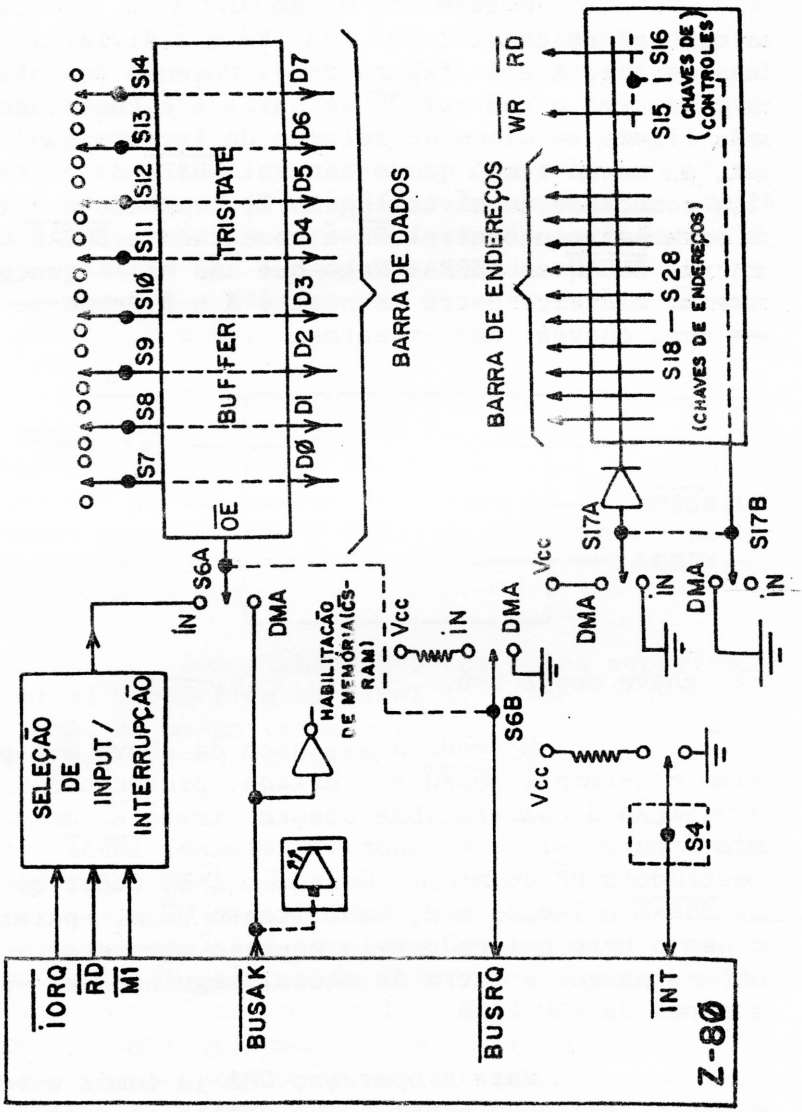
O microprocessador manipula os dados seguindo o programa monitor gravado na memória. O estágio de entrada de dados do MICRO-GALENA permite que o usuário introduza este programa através de uma operação denominada DMA, que grava diretamente na RAM. Além disto o estágio de entrada pode ser chaveado para operação IN (interrupção ou input), que copia dados do exterior em registro do microprocessador.

1 - ESQUEMA DE BLOCO DO ESTÁGIO DE ENTRADA DE DADOS

Na figura 3-1 vemos o esquema simplificado do estágio de entrada de dados, que será detalhado nas próximas seções.

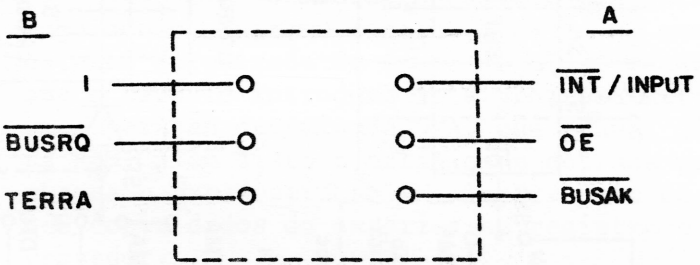
Os 8 bits de dados são chaveados pelo usuário, à sua livre escolha, nas chaves S7 até S14 e entram no byfer tristate como sinais IN0-IN7 (cada um em nível 1 ou 0).

Para que os bits IN0-IN7 atravessem o buffer e apareçam na barra de dados como D0-D7 é preciso que o terminal $\overline{OE}$ esteja habilitado - nível 0, caso contrário eles serão bloqueados e a saída permanecerá em tristate.



3-1 esquema simplificado do estágio de entrada de dados

O terminal $\overline{OE}$ do buffer é conectado ao central da chave dupla S-6. Esta é dividida em duas partes, A e B (figura 3-2). Movendo sua alavanca para cima o central $\overline{OE}$ da parte A é conectado ao pólo ligado ao bloco de seleção de interrupção/ input, ao mesmo tempo que o central $\overline{BUSRQ}$ da parte B fica conectado ao nível lógico 1; trazendo a alavanca para baixo o central $\overline{OE}$ é conectado a $\overline{BUSAK}$ e o central $\overline{BUSRQ}$ ao TERRA. Note que não há qualquer conexão elétrica entre as partes A e B, como se fossem duas chaves independentes.



3-2 chave dupla S-6

Movendo a alavanca da chave S-6 para baixo o terminal $\overline{BUSRQ}$ é aterrado, provocando uma requisição de barras (DMA-acesso direto à memória, veja figura 2-12), enquanto o terminal $\overline{BUSAK}$ fica conectado a $\overline{OE}$ do buffer .Quando o Z-80 ceder as barras $\overline{BUSAK}$ é levado a 0, habilitando $\overline{OE}$ e permitindo que o byte colocado pelo usuário atravesse o buffer e chegue à barra de dados, seguindo daí para os pinos da RAM 6116.

Para a operação DMA já temos a barra de dados, faltando endereços e controles para a RAM.

Com $\overline{BUSAK}$ em 0 automaticamente é habilitado o pino 18 $\overline{CS}$ da RAM (veja figura 1-12) e acende-se o LED indicador de DMA. Ao percebê-lo o usuário aciona a chave S-17 para DMA, fornecendo +5dcV às chaves de endereço S18-S28 através de diodos e terra para as chaves de controle S15 - S16. Com S17 na posição IN os diodos recebem zero Volt no anodo, ficando em corte e deixando as chaves S18 S28 em tristate (há 1 diodo para cada chave); ao mesmo tempo S15 e S16 não tem ligação com o terra para ativar $\overline{WR}$ e $\overline{RD}$.

O endereço desejado (na RAM) é assentado nas 11 chaves; quando $\overline{WR}$ for levado a 0 o byte colocado nas 8 chaves dos buffer será escrito na RAM (ou com $\overline{RD}$ = 0 será lido o byte do endereço existente nas chaves S18-S28). As 11 chaves permitem gravar ou ler em qualquer um dos 2.048 endereços internos da RAM.

Com a chave S-6 nos polos superiores o terminal $\overline{BUSRQ}$ fica em nível 1, desabilitando operação DMA. Ao mesmo tempo o terminal $\overline{OE}$ fica dependente do estágio de seleção de interrupção ou input.

A interrupção ocorre quando o usuário aterrar a chave S4, habilitando o terminal $\overline{INT}$ do Z-80. Este levará a nível 0 os terminais $\overline{IORQ}$ e $\overline{M1}$ (veja figura 2-13), que são logicamente combinados no bloco de seleção de interrupção/input para produzir nível 0 no terminal $\overline{OE}$ do buffer, habilitando-o. O byte colocado nas 8 chaves do buffer é lido pelo Z-80 (lembre-se que ele pode utilizá-lo de 3 maneiras diferentes, conforme estiver com os flags IM em IM0, IM1 ou IM2; veja Capítulo 2, seção 6, item INTERRUPÇÃO). Como a memória não parti-

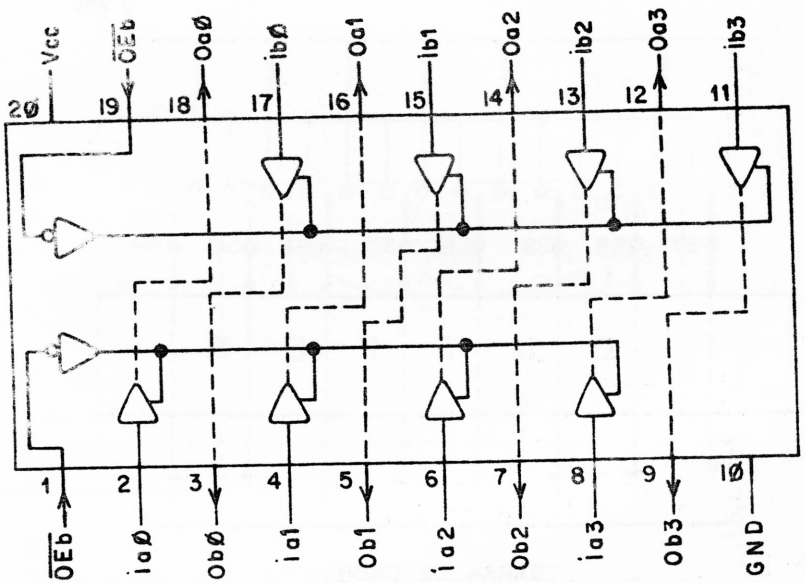
cipa da operação de interrupção a chave S-17 deve permanecer nos polos inferiores, cortando as 11 chaves de endereços e as chaves $\overline{WR}$ e $\overline{RD}$.

A operação INPUT (entrada) é ocasionada por uma instrução executada pelo Z-80. Ele leva a 0 os terminais $\overline{IORQ}$ e $\overline{RD}$ (veja figura 2-11A), ao mesmo tempo que coloca nos terminais A0-A7 o endereço da porta. Na configuração básica do MICRO-GALENA não foi atribuido nenhum endereço à porta IN de maneira que qualquer operação de entrada de porta ativará o circuito SELEÇÃO DE INTERRUPÇÃO/INPUT, habilitando o buffer.

$\overline{IORQ}$ e $\overline{RD}$ são combinados logicamente no bloco de seleção de interrupção/input, produzindo sinal nível 0 no terminal OE do buffer, habilitando-o. O byte colocado nas 8 chaves do buffer é lido pelo Z-80 e copiado no acumulador, sendo devidamente processado. Também na operação de INPUT a memória não é usada, devendo a chave S-17 ficar nos polos inferiores.

O buffer usado no estágio de entrada de dados é visto na figura 3-3. Na verdade ele é dividido em duas partes - a e b - de quatro bits cada. Levando a 0 o pino 1 - $\overline{OEa}$ ("output enable" a) as quatro bits de entrada da parte a (ia0, ia1, ia2, ia3) aparecem nas respectivas saídas (Oa0, Oa1, Oa2, Oa3); fazendo 0 o pino 19-$\overline{OEb}$ os quatro bits de entrada da parte b (ib0, ib1, ib2, ib3) ficam disponíveis nas respectivas saídas (Ob0, Ob1, Ob2, Ob3). Se $\overline{OEa}$ ou $\overline{OEb}$ estão em nível 1 as saídas correspondentes àquela parte permanecem em tristate.

No circuito do MICRO-GALENA os pinos 1 e 19 $\overline{OEa}$ e $\overline{OEb}$ são ligados juntos, para que se



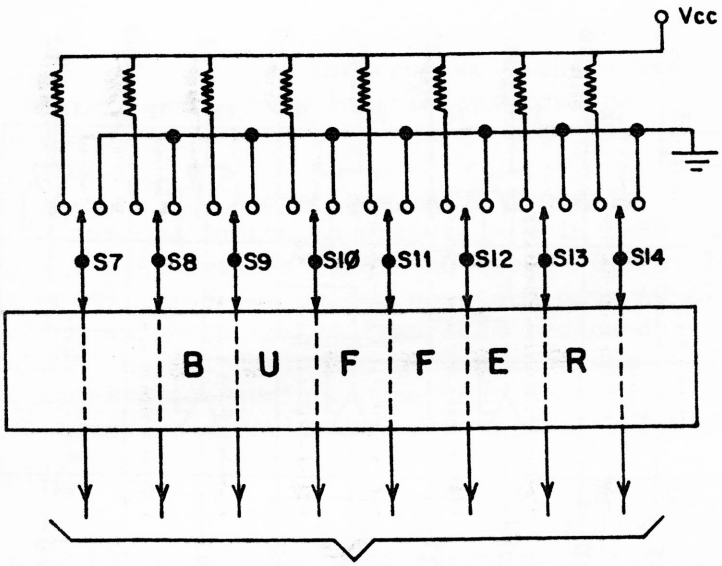
3-3 CI 74LS244 - buffer tristate

opere a entrada de 1 byte (4 bits de a + 4 bits de b). As entradas são conectadas às chaves manipuladas pelo usuário e as saídas à barra de dados. $\overline{OEa}$-$\overline{OEb}$ recebem o sinal DMA ou IN, conforme a posição da chave S-6A.

As 8 chaves disponíveis para o usuário dar entrada de dados podem estar em uma das duas posições: terra - nível 0 ou ligação a + 5dc V através de resistor de 4K7 Ohms - nível 1 (figura 3-4).

2 - OPERAÇÃO DMA

O circuito para operação DMA é mostrado na figura 3-5. Quando desejar executá-la o usuário pressiona a chave dupla S-6 para os polos

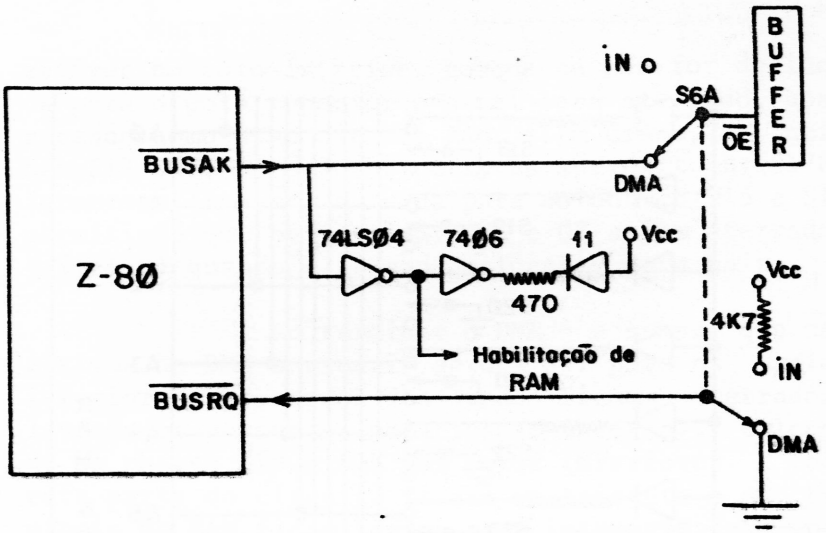


3-4 chaves de entrada de dados

inferiores, gerando sinal 0 no terminal $\overline{BUSRQ}$ do Z-80.

Ao final da instrução que estiver executando o microprocessador cede as barras de endereços (A0-A15), dados (D0-D7) e alguns controles ($\overline{MREQ}$, $\overline{IORQ}$, $\overline{RD}$, $\overline{WR}$), colocando-os em tristate. O usuário passa a controlá-los para entrada de dados diretamente na RAM.

A porta inversora 7406 antes do LED é um coletor aberto ("open colector" - oc): recebendo nível 1 na entrada transforma-o em 0 na saída, aterrando o resistor e drenando corrente da fonte; com nível 0 na entrada a saída deveria ser 1, mas a porta fecha-se em tristate, o que deixa toda a corrente da fonte (através do resistor) disponível para o próximo componente. No nosso circuito enquan

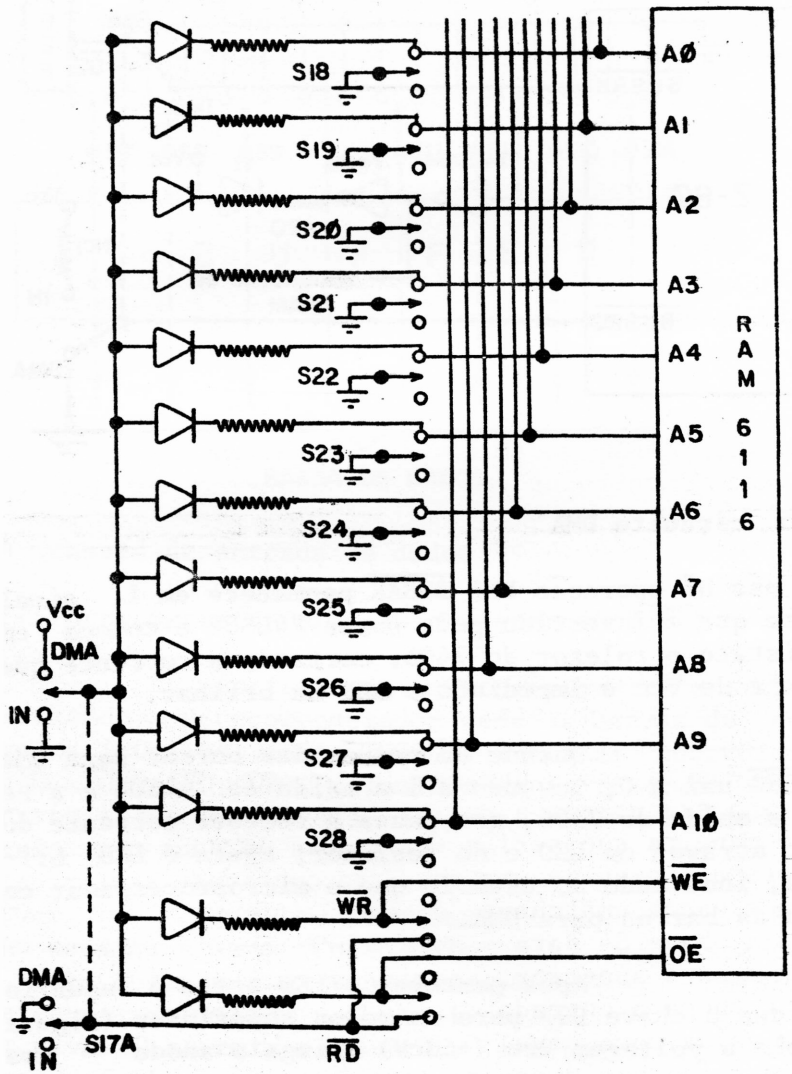


3-5 circuito DMA

to não há operação DMA $\overline{\text{BUSAK}}$ permanece em 1, sinal este que é invertido pela porta 74LS04 e coloca em tristate o coletor do 7406, cortando a corrente que viria de Vcc e impedindo o LED de brilhar.

Quando há cessão das barras para DMA $\overline{\text{BUSAK}}$ vai a 0, levando a 1 a saída do 74LS04 e a zero a saída do 7406, que passa a receber corrente do Vcc através do LED e do resistor; agora o LED brilha, indicando ao usuário que o microprocessador cedeu as barras para DMA.

Após perceber o LED aceso o usuário aciona a chave S17 para os polos superiores (figura 3-6): a voltagem Vcc (+5dcV) atinge o anodo dos diodos, atravessa-os e aos resistores, chegando aos pinos de endereços da RAM ainda como nível 1; isto se a respectiva chave de endereço (de S18 a S28)



3-6 chaveamento de endereços e controles na operação DMA

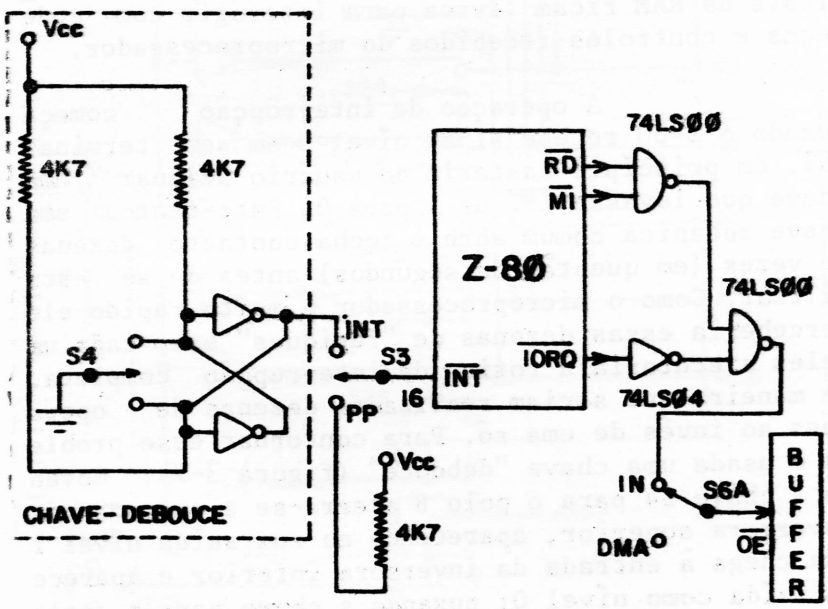
estiver no polo inferior, porque se ela for deslocada para o polo superior o sinal será aterrado, aparecendo como 0 no pino da RAM. Além disso, ao acionar S17 para os polos superiores sua parte B (S17B) fornecerá ligação ao terra para as chaves S15 e S16 permitindo que os terminais WE e OE sejam aterrados (nível 0) quando elas são deslocadas para baixo.

Ao terminar o DMA - e sempre que não estiver no DMA o usuário aciona S17 para os polos inferiores. Os anodos dos diodos ficam aterrados, levando-os ao corte (polarização inversa). Colocando as chaves S18 a S28 nos polos inferiores toda esta parte do circuito ficará em tristate e os terminais da RAM ficam livres para interagir com endereços e controles recebidos do microprocessador.

A operação de interrupção começa quando o Z-80 recebe sinal nível 0 em seu terminal $\overline{INT}$. Em princípio bastaria ao usuário acionar uma chave que levasse $\overline{INT}$ de 1 para 0. Entretanto, uma chave mecânica comum abre e fecha contacto dezenas de vezes (em questão de segundos) antes de se estabilizar. Como o microprocessador é muito rápido ele perceberia essas dezenas de "repiques" e em cada um deles executaria a rotina de interrupção completa, de maneira que seriam realizadas dezenas de operações ao invés de uma só. Para contornar esse problema é usada uma chave "debouce" (figura 3-7): movendo a chave S4 para o polo B aterra-se a entrada da inversora superior, aparecendo na sua saída nível 1 que chega à entrada da inversora inferior e aparece na saída como nível 0; puxando a chave para o polo A a saída da inversora inferior será 1, o que levará a saída da inversora superior a 0. O importante é que as saídas não mudam de estado enquanto a chave não encosta no outro polo, mantendo-se estável

nas transições e evitando "repiques". Portanto, para provocar uma interrupção no Z-80 basta acionar S4 do polo superior para o polo inferior.

O circuito da chave de debouce é aproveitado também no estágio de passo-a-passo. Com S3 no polo inferior o terminal $\overline{INT}$ do Z-80 fica em nível 1 ligado ao Vcc através de um resistor, independente da chave debouce, que passa a atuar no terminal de clock. Acionando S3 para cima a chave debouce será desligada do clock e ficará disponível exclusivamente para o terminal $\overline{INT}$, determinando seu estado.



3-7 circuito de interrupção

Ao receber nível 0 no pino $\overline{INT}$ o Z-80 leva a 0 os terminais $\overline{M1}$ e $\overline{IORQ}$ (veja figura 2-13), como reconhecimento da interrupção. $\overline{M1}$ é uma das entradas na porta NAND, que ao recebê-lo em nível 0 terá a saída em 1; $\overline{IORQ}$ em 0 é invertido. Assim, as duas entradas da segunda porta NAND estarão em 1 e como conseqüência sua saída será 0, habilitando o terminal $\overline{OE}$ do buffer. Observe que para $\overline{IORQ}$ ou as duas entradas $\overline{RD}$-$\overline{M1}$ da primeira NAND em 1 (desativados) a saída da segunda NAND será 1, desabilitando o buffer. Ou seja, com a chave S-6A no polo superior - operação IN - o buffer só será habilitado se $\overline{IORQ}$ = 0 e um dos terminais $\overline{RD}$ ou $\overline{M1}$ também for 0. $\overline{IORQ}$ = $\overline{M1}$ = 0 corresponde a interrupção, $\overline{IORQ}$ = $\overline{RD}$ = 0 refere-se a entrada de porta IO.

A operação acima descrita só ocorrerá se o Z-80 for previamente programado com a instrução "EI" ("enable interrupt", habilitação de interrupção), caso contrário ele não reconhecerá o pedido de interrupção feito no seu pino $\overline{INT}$. Lembre-se, ainda, que o Z-80 tem três modos diferentes de abordar a interrupção, selecionados pelas instruções "IM0", "IM1" e "IM2".

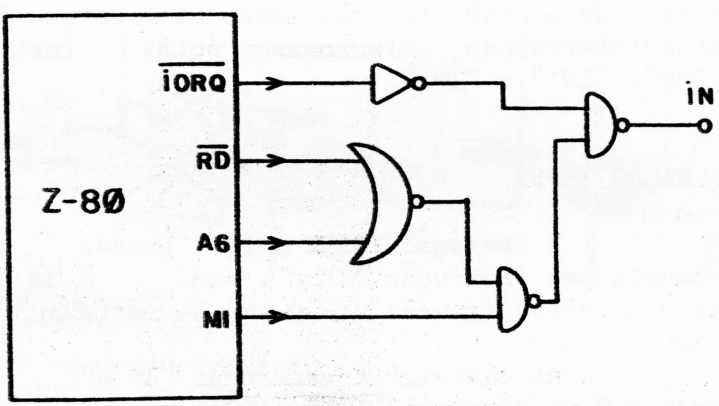
3 - OPERAÇÃO INPUT

A operação INPUT ocorre quando o Z-80 executa uma instrução "IN n", onde 'n' é um número de 0 a 255 designando uma porta de entrada.

Ao executar a instrução "IN n" o Z-80 leva a 0 os terminais $\overline{IORQ}$ e $\overline{RD}$, ao mesmo tempo que coloca na barra de endereços os 8 bits correspondentes a n (veja figura 2-11A).

Com $\overline{IORQ} = \overline{RD} = 0$ o terminal $\overline{OE}$ do buffer é habilitado, exatamente como ilustrado na figura 3-7 para operação de interrupção (a única diferença é que $\overline{RD}$ vai a 0 enquanto $\overline{MI}$ permanece em 1, mas de qualquer maneira a saída da primeira NAND será 1). Os dados chaveados nas 8 chaves do buffer serão copiados no acumulador do microprocessador e devidamente processados.

Com o esquema acima o número n da porta é desconhecido e qualquer que ele seja haverá habilitação do buffer. Se for necessário "numerar" a porta de entrada, designando-lhe uma posição, o terminal $\overline{RD}$ é combinado numa porta NOR com o terminal de endereço desejado antes da entrada na primeira NAND. Na figura 3-8 apresentamos um exemplo utilizando A6. A primeira NAND só receberá sinal 1 na entrada superior quando $\overline{RD}$ = A6 = 0. Ou seja, o endereço da porta será n = X0XX, XXXX, onde X pode ser 1 ou 0.



3-8 numeração de porta de entrada (não é usada no MICRO-GALENA).

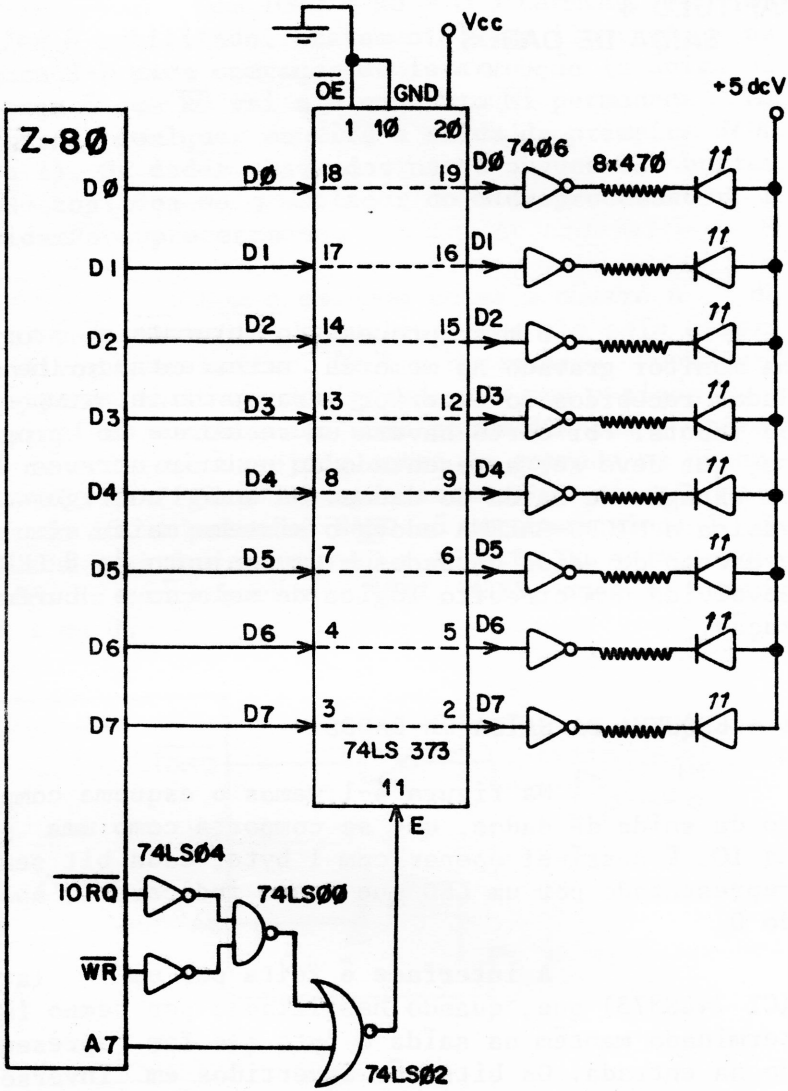
CAPÍTULO 4
SAÍDA DE DADOS

O microprocessador executa o programa monitor gravado na memória, acrescentando-lhe os dados recebidos do exterior através de interrupções ou inputs. Por certo haverá um resultado do programa, que deve ser apresentado ao usuário através de um estágio de saída de dados. Na sua configuração básica o MICRO-GALENA adota o sistema mais simples e direto de saída de dados - um conjunto de 8 LEDs, envolvido num circuito lógico de seleção e bufferização.

1 - ESQUEMA DA SAÍDA DE DADOS

Na figura 4-1 vemos o esquema completo da saída de dados, que se comporta como uma porta IO. É possível operar com 1 byte, cada bit sendo representado por um LED que aceso indicará 1, apagado 0.

A interface é feita por um latch (CI 74LS373) que, quando habilitado, por tempo indeterminado mantém na saída o byte que for apresentado na entrada. Os bits são invertidos em inversora coletor aberto e controlam o estado dos LEDs.



4-1 esquema da saída de dados

2 - HABILITAÇÃO DA SAÍDA DE DADOS

A operação de saída de dados por porta IO decorre de uma instrução "OUT n" executada pelo microprocessador, onde 'n' é o endereço da porta. O Z-80 leva a 0 os terminais $\overline{IORQ}$ e $\overline{WR}$, colocando na barra de endereços o byte n (veja figura 2-11 B). A diferença da entrada de dados é que $\overline{WR}$ vai a 0 enquanto $\overline{RD}$ permanece em 1.

No MICRO-GALENA $\overline{IORQ}$ e $\overline{WR}$ são invertidos e combinados numa NAND; assim, esta só terá 0 na saída quanto $\overline{IORQ}$ = $\overline{WR}$ = 0. A saída da NAND é combinada com A7 numa NOR, que só terá 1 na saída quando essas duas entradas estiverem em 0. Portanto, a condição para que a saída da NOR seja 1 - habilitando o terminal E ("enable") do buffer - é $\overline{IORQ}$ = $\overline{WR}$ = A7 = 0. Quando estes três terminais estiverem simultaneamente em 0 o byte será habilitado, apresentando e mantendo na saída o byte que estiver na entrada.

Em conseqüência esta porta de saída estará na posição n = 0XXX XXXX, onde o 0 do MSB corresponde a A7 e os sete 'x' podem ser 1 ou 0, indiferentemente, dos bits A0 até A6. Ao executar a instrução "OUT 0XXX XXXX" o Z-80 fará $\overline{IORQ}$ = $\overline{WR}$ = A7 = 0, provocando sinal 1 no terminal E de habilitação do latch, ao mesmo tempo que o byte a ser apresentado é posto na barra de dados.

3 - LATCH 74LS373

O latch 74LS373 precisa de sinal nível 1 no pino 11-E ("enable", habilita) para ser habilitado. O byte que estiver na sua entrada no ins-

tante da habilitação é gravado e permanecerá na saída até que for alterado por outra operação (veja que esta é a diferença entre latch e buffer; o segundo não mantém os dados quando termina a habilitação).

Além de E o latch 74LS373 tem o terminal de controle $\overline{OE}$ ("output enable", habilita saída). Quando este está em nível 1 os 8 pinos de saída permanecem em tristate; em nível 0 eles são liberados para apresentação dos bits. No MICRO-GALENA o pino 1-OE do 74LS373 é mantido aterrado, habilitando permanentemente a saída.

4 - APRESENTAÇÃO DOS DADOS

A apresentação dos dados é feita por LEDs, correspondendo à forma binária. LED aceso representa bit 1 e LED apagado bit 0. Eles são usados para apresentar o resultado do programa introduzido pelo usuário.

Quando o bit na saída do latch for 1 provocará nível 0 na saída da inversora, fazendo a corrente originada na fonte +5dcV atravessar o LED - acendendo-o - e o resistor limitador de corrente, sendo absorvida no CI 7406. Para bit 0 na saída do latch a saída da inversora coletor aberto é cortada, impedindo que haja circulação de corrente e mantendo o LED apagado.

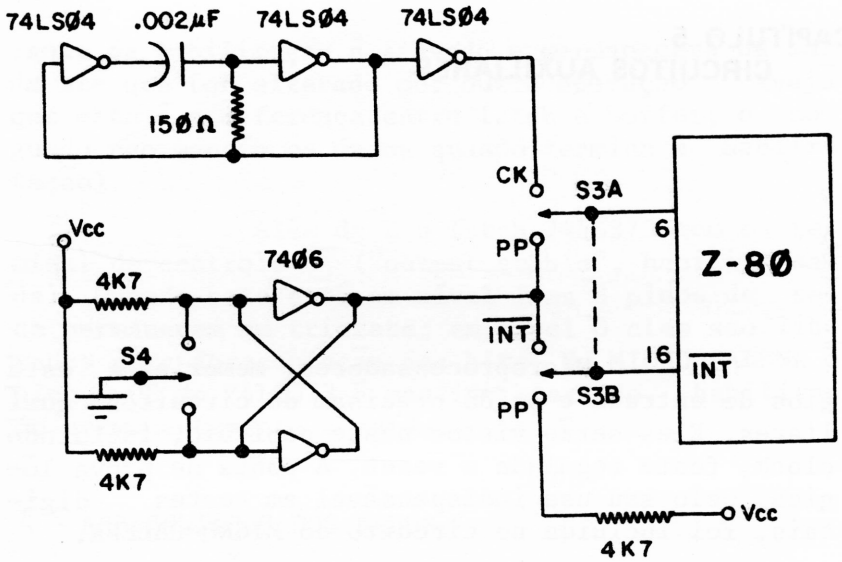
CAPÍTULO 5
CIRCUITOS AUXILIARES

Microprocessadores, memórias e estágios de entrada e saída precisam de circuitos auxiliares. Eles serão vistos neste capítulo, incluindo clock, fonte regulada e reset. A ponta de prova lógica, pelo seu uso indispensável em testes digitais, foi incluída no circuito do MICRO-GALENA.

1 - CIRCUITO CLOCK X PASSO-A-PASSO

O MICRO-GALENA permite a sincronização do Z-80 de duas maneiras: clock ou passo-a-passo. A primeira é usada na operação normal do computador, dispensando a intervenção do usuário; a segunda é controlada manualmente e empregada quando se quer estudar em detalhes uma parte da execução do programa.

Com a chave S3 nos polos superiores o terminal 6-CLOCK do Z-80 recebe o sinal saindo da inversora, sendo sincronizado por ele - a cada pulso a execução do programa é avançada mais um passo. A inversora é um buffer de saída para o circuito oscilador precedente, cuja freqüência é aproximadamente F = 1/1,4 x R x C, onde R é o valor do resistor em Ohms e C do capacitor em Farads. Substituindo pelos valores da figura 5-1:



5-1 circuito de clock x passo-a-passo

$F = 1/1,4 \times 150 \times 0,002 \times 10^{-6}$, aproximadamente F = 2MHZ.

Entretanto, este valor de freqüência do clock é impreciso, pois no circuito real as chaves e ligações acrescentam uma capacitância apreciável. De qualquer forma representa um valor conservador, mantendo uma larga margem de segurança, pois o Z-80A admite até 4MHZ (neste nível até o Z-80, que tem o máximo de 2,5 MHZ, pode ser empregado).

Acionando a chave S3 para os polos inferiores o oscilador é desligado do terminal 6-CLOCK, e este passa a ser sincronizado pela chave debouce anteriormente conectada ao pino de interrupção. Agora, um "pulso" de clock será um movimento manual da chave S4 de baixo para cima e voltando à

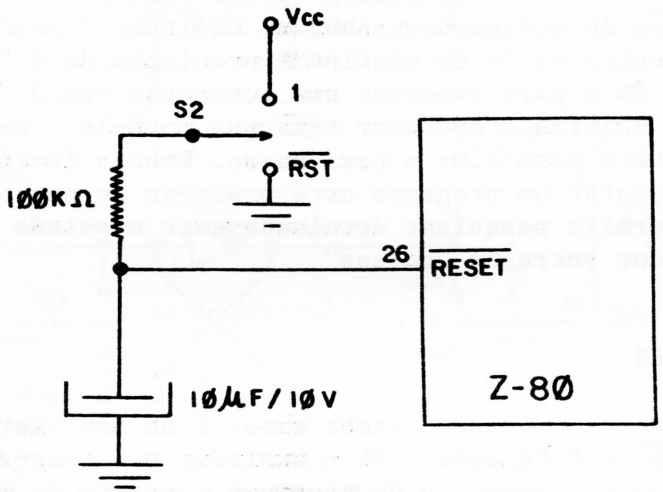
posição anterior. Verificando pelas figuras de temporização do microprocessador no Capítulo 2 percebe que um único ciclo de máquina M precisará de 4 "pulsos" de S4 e para executar uma instrução com 3 ciclos de máquina o operador terá que movê-la cerca de 12 vezes para cima e para baixo. Embora inviável para executar um programa este processo de passo-a-passo permite pesquisar detalhadamente o estado do computador entre os "pulsos".

2 - RESET

Quando recebe sinal 0 no seu terminal $\overline{\text{RESET}}$ o Z-80 zera o PC - contador de programa, reiniciando a execução do programa a partir da instrução gravada no endereço 0 e seguindo em ordem crescente. Para que ocorra efetivamente o reset é necessário que o terminal $\overline{\text{RESET}}$ seja mantido em nível 0 durante pelo menos 6 ciclos de clock.

O circuito de reset do MICRO - GALENA é visto na figura 5-2. Com a chave S2 no polo superior o capacitor se carrega até atingir +5 Volts, mantendo o terminal 26 - $\overline{\text{RESET}}$ do Z-80 em nível 1. Acionando a chave para o polo inferior $\overline{\text{RST}}$ seu central é aterrado, descarregando o capacitor através do resistor de 100 K Ohms e provocando nível 0 no terminal 26 - $\overline{\text{RESET}}$ do Z-80.

O tempo necessário para carregar ou descarregar o capacitor depende da constante T = R x C, onde C é o valor do próprio capacitor e R do resistor. Propositalmente foram escolhidos valores grandes (100 K Ohms e 10 micro Farads, de modo a diminuir os efeitos de repique da chave S2. O usuário precisa aguardar alguns segundos até que se com



5-2 circuito de reset

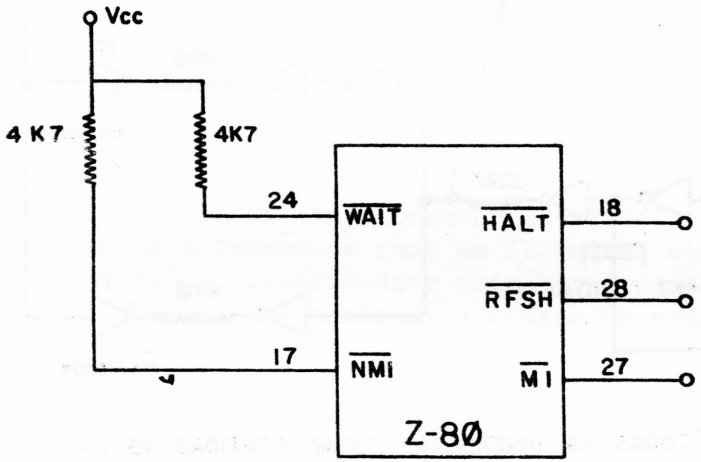
plete a carga ou descarga e o Z-80 saia do ciclo de reset ou entre nele, respectivamente.

3 - TERMINAIS DESATIVADOS DO MICROPROCESSADOR

Terminais de entrada não usados não podem ficar desconectados, pois seriam perdebidos como níveis lógicos, distorcendo a lógica do circuito.

No MICRO-GALENA os terminais de entrada no microprocessador WAIT e NM1 não são usados, sendo conectados ao Vcc através de resistores. Portanto, desativados com nível 1.

Quanto a terminais de saída não geram problemas (de lógica) quando desconectados. No



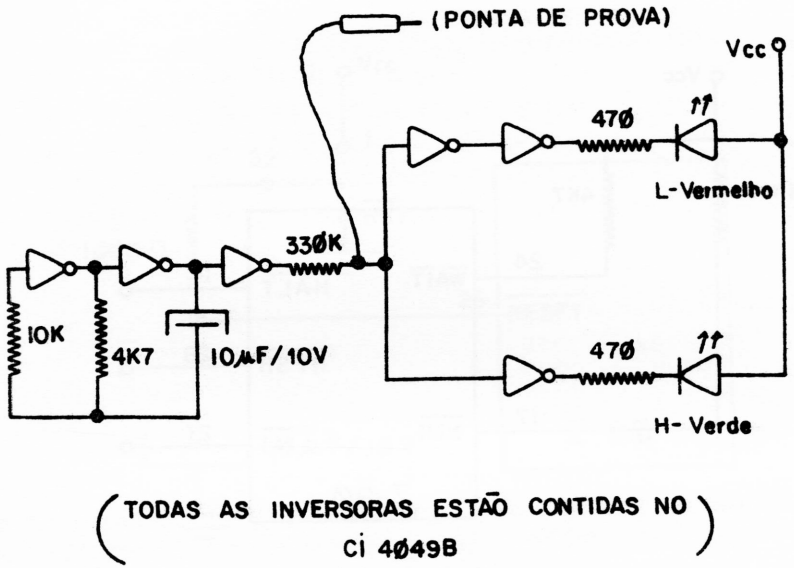
5-3 terminais desativados do microprocessador

MICRO-GALENA os terminais de saída do microprocessador HALT e RFSH não são usados e permanecem flutuando (sem ligação com o circuito).

4 - PONTA DE PROVA LÓGICA

A ponta de prova lógica apresentada na figura 5-6 na realidade não faz parte do MICRO GALENA; o usuário que dispuser de uma similar poderá omiti-la. Mas seu uso é tão indispensável que ela foi incluida.

As duas inversoras à esquerda formam um oscilador que fazem os dois LEDs piscarem alternadamente. Quando a ponta de prova é encostada num ponto com nível 1 o sinal é invertido pela in-



5-4 circuito da ponta de prova lógica

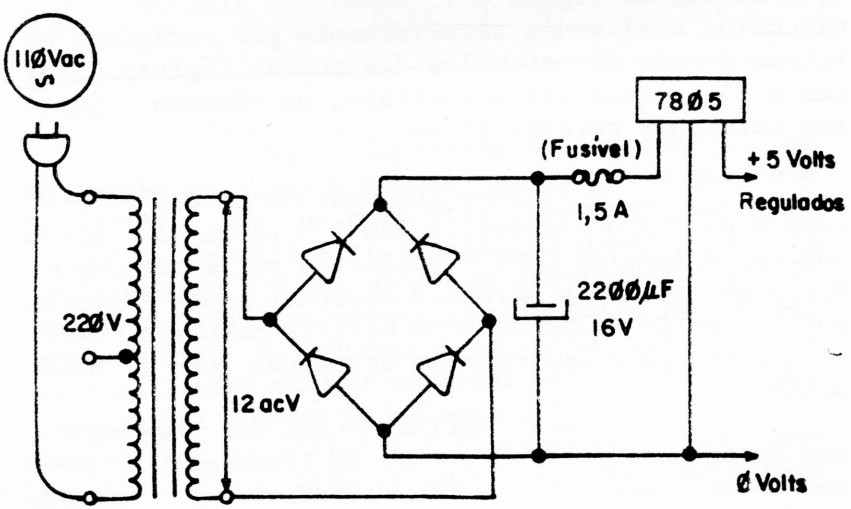
versora inferior e aparece como nível 0 na sua saída, drenando corrente do LED-H (verde) e fazendo-o brilhar continuamente; ao mesmo tempo esse sinal é invertido e "desinvertido" nas inversoras superiores, aparecendo como nível 1 no catodo do LED-1 (vermelho), mantendo-o apagado. Por outro lado, com a ponta de prova num ponto com nível 0 o sinal aparecerá como 1 no catodo no LED-H, mantendo-o apagado, e como 0 no catado do LED-L, fazendo-o brilhar constantemente. Com a ponta de prova encostada num ponto em tristate o oscilador continua a alimentar os LEDs normalmente, fazendo-os piscarem alternadamente. Em resumo: (1) ponta de prova em 0: LED-L aceso e LED-H apagado (2) ponta de prova em 1: LED - H aceso e LED-L apagado (3) ponta de prova em tristate: os dois LEDs piscando alternadamente.

5 - FONTE REGULADA

O esquema da fonte regulada +5dc V é visto na figura 5-5. Sua função é transformar a tensão 110V ou 220V alternados da tomada em 5 Volts dc regulados com tolerância de 5% (4,75V a 5,25V).

Um transformador 110-220/12-1 Ampére transforma a tensão da rede em 12 Volts, ainda ac (alternada). O usuário deve escolher o ponto de ligação 110V ou 220V conforme a tensão da rede na cidade em que mora.

A tensão de 12 V na saída do transformador é retificada por uma ponte de diodos, tornando-se dc, carregando um capacitor filtro. A seguir temos um fusível de proteção de 1,5 Ampére. O regulador de tensão 7805 incrustado num dissipador de calor regula a tensão para +5Volts dc, independente da carga (até certo limite).



5-5 Fonte regulada +5Volts dc

CAPÍTULO 6
ESQUEMA ELÉTRICO

Neste capítulo veremos o esquema elétrico do MICRO-GALENA e suas características elétricas. Mostraremos também as expansões de memória e porta de saída previstas na configuração básica e já constando no esquema.

1 - CARACTERÍSTICAS ELÉTRICAS

O esquema elétrico do MICRO-GALENA é apresentado na figura 6-1, sendo uma síntese dos circuitos analisados anteriormente por estágios. As letras dentro dos símbolos das portas lógicas indicam o CI no qual estão contidas; os números junto aos terminais referem-se aos pinos do CI.

O sistema foi baseado em CIs TTL - exceto o microprocessador, a RAM e a ponta da prova lógica. A família CMOS é preferida em microcomputadores por ser mais barata e consumir menos energia, porém é muito delicada e de difícil manuseio, defeito capital para um trabalho artesanal como o MICRO GALENA.

A sub-família 74LSXX foi empregada, com exceção dos dois CIs 7406 (6 inversoras - coletor aberto) que são fabricados somente no modelo

"standard". Essa opção mantém um bom equilíbrio entre consumo de energia e velocidade.

As especificações elétricas do Z-80 e da família TTL standard e LS são registradas na tabela abaixo, servindo de referência para a discussão a seguir. A figura 6-2 ilustra o sentido da corrente nos dois níveis, servindo tanto para o Z-80 como para o TTL.

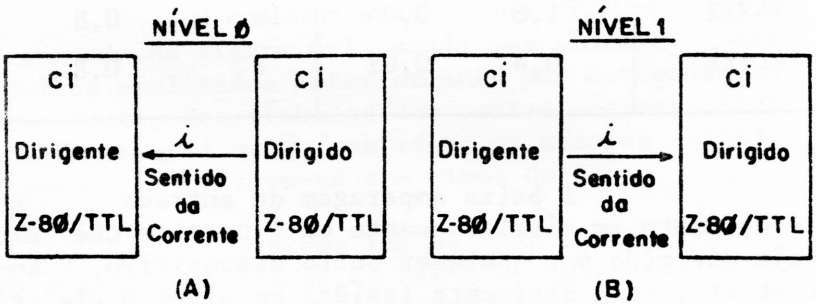
TABELA DE ESPECIFICAÇÕES ELÉTRICAS DO Z-80 E TTL

CI	ENTRADA (mA)		SAÍDA (mA)	
	0	1	0	1
Z-80	0,01	0,01	1,8	0,25
74XXX	1,6	0,04	16	0,8
74LSXX	0,4	0,01	8	0,8

A baixa amperagem de entrada do Z-80, tanto em nível 0 quanto em 1 permite que ele seja dirigido por qualquer outro dispositivo. Entretanto, como dirigente (saída) em nível 0 ele recebe no máximo 1,8 mA; assim só poderá ser ligado a um único 74XX, que como dirigido emite no mínimo 1,6 mA em nível 0. Digamos que o Z-80 controle um 74XX e um 74LSXX no seu terminal $\overline{MREQ}$: quando ele quizer ativar a memória baixará a 0 Volt esse pino, recebendo ali no máximo 1,8 mA originados nos dois CIs dirigidos; o 74XX responderá enviando no mínimo

1,6 mA e o 74LSXX no mínimo 0,4 mA, o que dará no mínimo 2,0 mA, amperagem maior que o Z-80 pode absorver em nível 0.

A discussão acima visa alertar o leitor que ao construir seu MICRO-GALENA deve seguir rigorosamente as especificações de subfamílias, caso contrário o sistema não funcionará. Ao expandi-lo esse assunto também deve ser levado em consideração, conectando o Z-80 com CIs CMOS ou 74LSXX e evitando ligações diretas com o 74XX.



6-2 sentido da corrente para sinal (a) nível 0 (b) nível 1.

2 - EXPANSÃO DE MEMÓRIA

A expansão de uma memória RAM-6116 - de 2KBytes é a mais imediata no MICRO-GALENA. Na placa é previamente reservado um soquete para o novo CI 6116 e os componentes empregados na lógica da expansão já estão disponíveis na configuração básica. É só encaixar o CI de expansão e passar a usá-lo.

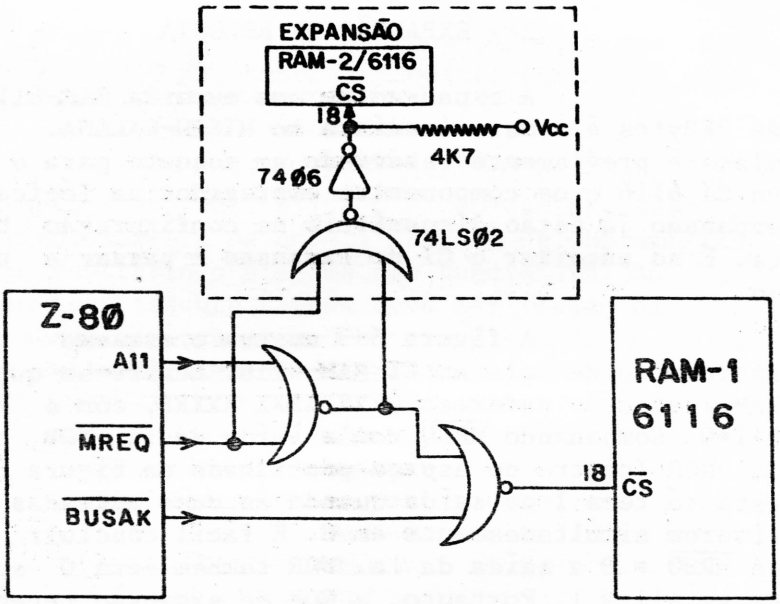
A figura 6-3 mostra o esquema para introdução de mais um CI RAM-6116. Lembre-se que a RAM-1 está no endereço 0XXX XXXX XXXXB, com o MSB= A11=0. Combinando $\overline{MREQ}$ com a saída da 1a. NOR numa 3a. NOR (dentro do espaço pontilhado na figura 6-3), esta só terá 1 na saída quando as duas entradas estiverem simultaneamente em 0. É fácil concluir que se $\overline{MREQ}$ = 0 a saída da 1a. NOR também será 0 somente se A11 = 1. Portanto, a NOR de expansão receberá as duas entradas em 0 quando $\overline{MREQ}$ = 0 e A11 = 1;neste caso sua saída é 1 que, invertido pela porta do 7406, aparece como 0 no pino 18-$\overline{CS}$ do 2º CI RAM-6116, habilitando-o. Seu endereço é 1XXX XXXX XXXX B, indo de 1000 0000 0000B = 2048 até 1111 1111 1111B = 4095.

Observe pela figura 6-1 que tanto a NOR quanto a inversora 7406 usadas na expansão da memória são "sobras" da configuração básica.

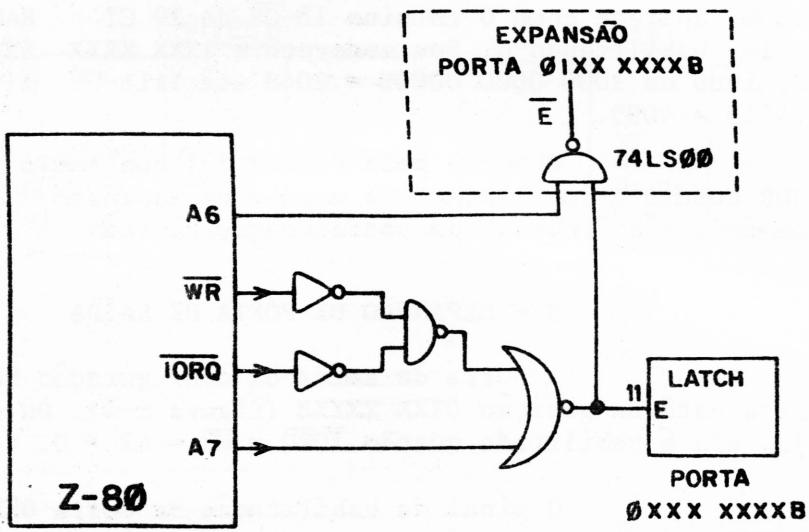
3 - EXPANSÃO DE PORTA DE SAÍDA

A porta de saída da configuração básica está na posição 0XXX XXXXB (figura 6-4). Ou seja, ela é habilitada quando $\overline{IORQ}$ = $\overline{WR}$ = A7 = 0.

O sinal de habilitação da porta 0XXX XXXXB pode ser aproveitado para expansão de mais



6-3 expansão de memória RAM



6-4 expansão de porta de saída

uma porta de saída, combinando-o com A6 numa NAND (dentro do espaço pontilhado na figura 6-4), que só apresentará $\overline{E}$=0 na saída quando $\overline{WR}$ = $\overline{IORQ}$ = A7 = 0 e A6 = 1. A posição dessa 2a. porta de saída será 01XX XXXXB.

Um detalhe interessante é que a habilitação da porta de expansão 01XX XXXXB sempre acarretará a ativação simultânea da porta 0XXX XXXXB, mas o inverso não é verdadeiro.

A porta de expansão poderá ser projetada como na configuração básica, o sinal $\overline{E}$ servindo para habilitar (lógica negativa) um latch que receba o byte de dados. Mas $\overline{E}$ também pode ser empregado sozinho, por exemplo excitando um alto-falante ou um circuito que excite o alto-falante.

A porta NAND usada nesta expansão é outra "sobra" da configuração básica, já disponível no circuito original.

OBS.
A figura 6-1 encontra-se no final do livro.

CAPÍTULO 7
DESCRIÇÃO FÍSICA DO MICRO GALENA

Até agora vimos a descrição elétrica do MICRO-GALENA, incluindo a teoria de funcionamento. Neste capítulo veremos sua descrição física, contendo três blocos: painel, fonte de alimentação e ponta de prova.

1 - BLOCOS DO MICRO-GALENA

O MICRO-GALENA pode ser dividido em 3 blocos principais: painel, fonte e ponta de prova lógica. Na verdade apenas o painel é indispensável ao projeto, os outros dois dependendo dos recursos e objetivos do usuário.

O painel contém os estágios do MICRO GALENA: microprocessador, memória, circuitos lógicos, chaveamentos e LEDs indicadores; inclui ainda o regulador de tensão com o respectivo dissipador de calor.

Considerando-se que o MICRO-GALENA é projetado prioritariamente para fins didáticos a fonte foi deixada aparte. Isso devido ao peso e tamanho do transformador, além dos riscos inerentes aos circuitos ligados diretamente à tomada de força 110 V/220V da rede. Colocando-a separadamente fica mais fácil e seguro manipular o painel para fins de tes-

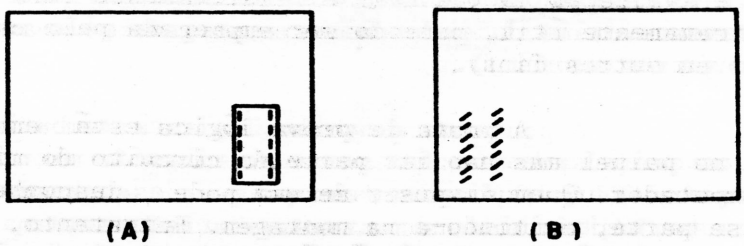
te e avaliação (afora isso é relativamente cara e extremamente útil, podendo ser empregada pelo usuário em outros fins).

A ponta de prova lógica está embutida no painel mas não faz parte do circuito do microcomputador. Quem dispuser de uma pode desconhecer essa parte, omitindo-a na montagem. Entretanto, para fins didáticos ela é tão útil e indispensável que vale a pena tê-la permanentemente junto ao resto do circuito.

2 - PAINEL

Na figura 7-1 é visto o painel do MICRO-GALENA. Consiste numa placa de madeira de 30 cm x 21 cm, com 4 cm de espessura, montada sobre 4 suportes laterais de 3 cm de altura, parafusadas às suas bordas.

Os soquetes de CIs, em número de 11, são fixados ao painel de tal maneira que no lado superior ficam no mesmo nível da placa, com os encaixes ("buraquinhos") para colocação do respectivo CI (figura 7-2A); no lado inferior da placa aparecem os pinos do soquete, disponíveis para as soldagens e ligações (figura 7-2B). Com o micro já montado o usuário trabalha no lado superior da placa e sem perceber o soquete, como se os CIs estivessem encaixados diretamente na madeira; virando o console verá o lado inferior dos soquetes com os pinos sobressaindo para as ligações. Os soquetes têm aproximadamente 3 cm de altura - afora os pinos - e se a placa também tiver espessura pouco maior eles ficarão perfeitamente embutidos. Pequenas diferenças, de até 1 cm na espessura da placa ou na altura dos



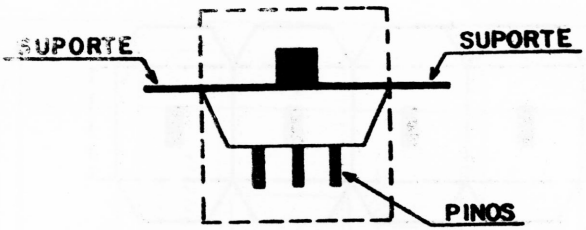
7-2 posição do soquete na placa (A) lado superior-encaixes (B) lado inferior-pinos

soquetes não prejudicam o projeto.

Na sua configuração básica o MICRO GALENA usa apenas 1 CI de memória RAM, com 2 K bytes. Para permitir fácil expansão de mais 2 K bytes um segundo soquete é fixado na placa, embora não seja utilizado inicialmente.

As chaves estão divididas em três grupos - dados, endereços e controles. Todas são do tipo H-H (liga-desliga) e devem seguir o esquema da figura 7-3, com dois suportes de fixação e pinos na parte inferior. No painel é feita uma fenda do tamanho do corpo da chave (parte pontilhada da figura 7-3), de maneira que os dois suportes encostem na parte superior da placa, sendo colados ali, enquanto os pinos aparecem no lado inferior junto com a pinagem dos soquetes de CIs. Na parte superior a alavanca da chave é manipulada pelo usuário, enquanto na inferior são feitas as soldagens e ligações.

Seria muito trabalhoso fazer uma fenda na placa para cada chave, o que resultaria ainda



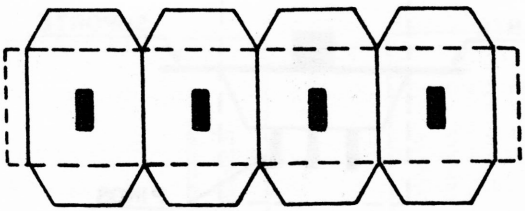
7-3 chave H-H

em maior tamanho e fragilidade do painel. Por esse motivo elas são fixadas em paralelo, aproveitando uma única fenda na placa para cada bloco de chaves (figura 7-4).

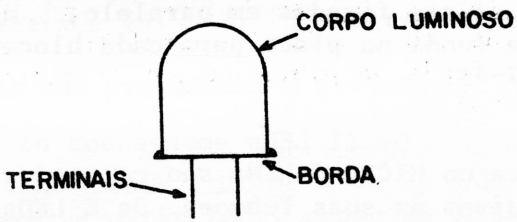
Os 11 LEDs empregados na configuração básica do MICRO-GALENA são colocados em posições alusivas às suas funções. Os 8 LEDs de saída de dados ficam na parte superior esquerda; os dois da ponta de ponta também à esquerda no meio, um verde e outro vermelho; entre as chaves S-6 de pedido de DMA e S-17 de liberação das chaves de dados para DMA fica o LED indicativo de $\overline{\text{BUSAK}}$ (figura 7-1).

O LED tem seus dois terminais saindo do corpo luminoso (figura 7-5), com bordas "sobrando" dos pontos de tirada. No MICRO-GALENA uma fenda maior que o tamanho da distância entre os dois terminais do LED e menor que seu corpo luminoso é feita no console (figura 7-6), de maneira que a borda se apoie na placa, com o corpo luminoso visível na parte superior e os terminais disponíveis na inferior.

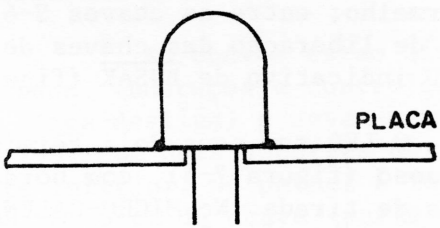
O regulador de tensão é parafusado ao dissipador de calor e os dois fixados no canto superior direto do painel no lado de cima. Seus



7-4 4 chaves fixadas em paralelo. A linha pontilhada é a fenda na placa.



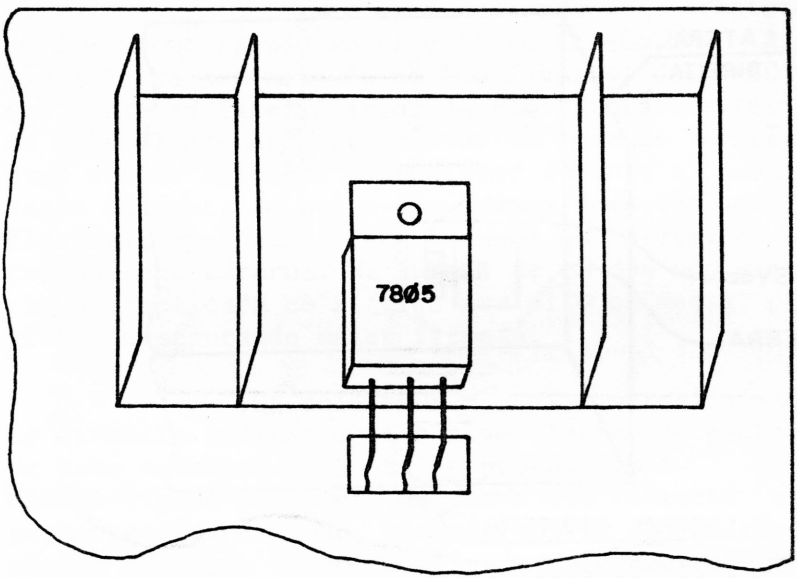
7-5 descrição física do LED



7-6 fixação do LED à placa

três terminais atravessam uma fenda e ficam dispo níveis no lado inferior para as devidas ligações (figura 7-7).

A face inferior do painel (lado ave so da figura 7-1) constará de pinos dos soquetes e das chaves, além dos demais componentes do cir cuito (resistores, capacitores, diodos). As liga-

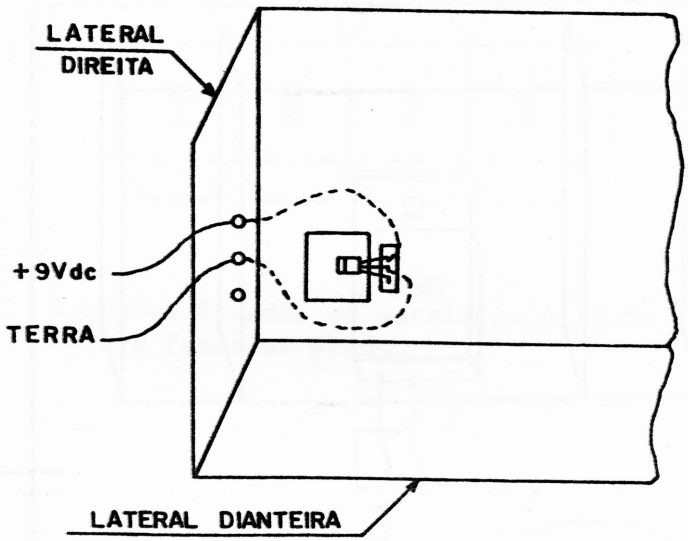


7-7 fixação do regulador de tensão com dissipador na placa

ções são feitas através de fio rígido número 24 ou maior (quanto maior o número menor o fio), encapado.

O percurso da fiação - que será apresentado adiante - foi extensamente pesquisado para evitar problemas de impedância e interferência. É recomendado ao leitor seguir estritamente o mapa de fiação e em caso de expansão ou alteração usar o menor percurso possível.

Na lateral direita do suporte são feitos três orifícios nos quais são colados fios rígidos grossos ou hastes metálicas, tal que cada um tenha uma extremidade sobressaindo para o exterior e outra dentro do painel, próximo à fenda que recebe os três pinos do regulador de tensão (figura 7-8). O primeiro fio ou haste é ligado ao primeiro



7-8 ligação da fonte ao painel

pino (IN) do regulador e sua extremidade no exterior fica disponível para ser conectada aos +9 dc V da fonte através de um "jacaré". Igualmente, o segundo fio ou haste é ligado ao segundo pino (REF) do regulador e sua extremidade no exterior fica disponível para ser conectada ao terra da fonte através de um "jacaré". O terceiro fio ou haste é ligado ao terceiro pino do regulador (+ 5 Volts REGULADOS) e não é conectado ao exterior. As extremidades internas dos fios ou hastes, dentro do painel, além das ligações com o regulador de tensão também são tomadas como origem de terra e Vcc (+ 5 Volts REGULADOS) para o resto do circuito.

3 - PONTA DE PROVA LÓGICA

Uma ponta de prova semelhante àque-

las usadas em multímetros é retirada do resistor de 330 KOhms ligado ao pino 15 do CI 4049. A ligação é feita por meio de um fio flexível de 50 cm que passa da face inferior da placa para o exterior da lateral esquerda através de um pequeno orifício. Para evitar agitação do resistor durante a manipulação da ponta de prova são feitos dois nós no fio flexível, um junto à face interna da lateral e outro na face externa. Na junção de um dos nós com a placa é aplicada cola (tipo Araldite ou massa plástica), assegurando maior firmeza.

Os dois LEDs - azul e vermelho - deste circuito indicam qual o nível lógico do ponto onde está encostada a ponta de prova. Com o MICRO GALENA ligado e a ponta de prova sem contacto eles permanecerão piscando, podendo servir de referência para a condição ligado/desligado do computador.

4 - FONTE DE ALIMENTAÇÃO

A fonte de alimentação terá por função transformar a tensão alternada da rede, 110 Volts ou 220 Volts, em tensão contínua entre 9 Volts ainda não regulados, que serão entregues ao regulador de tensão.

Este estágio é montado sobre uma placa de madeira de 7 cm x 18 cm, sustentada por quatro suportes (pés). Os dois fios da tomada de ligação à rede entram por baixo da placa e aparecem na sua face superior através de fendas (figura 7-9); nesses pontos são dados nós nos fios (representados na figura por pontos cheios expandidos) para melhorar a fixação. Os dois fios devem ser ligados ao primário do transformador (110 Volts ou 220 Volts, conforme a especificação na cidade do usuário). En

tretanto, um deles passa antes por uma chave (correspondendo a S1 no esquema elétrico), cujo central completa a conexão com o transformador. Essa é, portanto, a chave liga/desliga da fonte de alimentação (e do computador).

O transformador dispõe de dois suportes laterais com fendas, nas quais se encaixam parafusos que são fixados por porcas enroscadas na face inferior. Os dois fios do secundário (9 Volts ou 12 Volts, ainda ac) são levados à face inferior, retornando à superior através de duas fendas. Nesses locais são dados nós nos fios e a posição fixada com cola ou massa plástica, de maneira que possam ser usados como pontos fixos pela ponte de diodos, sem risco de deslocamento.

Um fio preto de 50 cm é levado por baixo da placa e aparece na face superior através de uma fenda, recebendo ali um nó e cola. Ele será o terra (GND, zero Volt) que se ligará por meio de um jacaré ao painel.

Temos agora três pontos fixos, com respectivos fios dispondo de pequenas pontas descascadas prontas para receberem as ligações da ponte de diodos.

Estes são soldados às pontas, formando um quadrado fixo e suspenso cerca de 1/2 cm acima da placa. Note que é preciso colar muito bem à placa os três nós de onde saem as pontas (terra-fio preto e os dois fios do secundário do transformador), caso contrário os diodos mexerão e desfarão a soldagem que os mantém unidos.

O capacitor de filtragem é colado "em pé" na placa com o lado de onde saem os dois terminais voltado para cima. O negativo é soldado através de um fio rígido ao terra da ponte de diodos (encontro dos dois anodos). O positivo vai, ainda por meio de fio rígido, à saída da ponte (encontro dos dois catodos).

A saída positiva da ponte de diodos não é soldada a qualquer nó colado à placa. A ponta do terminal de um dos diodos é ligeiramente curvada para cima e soldada à uma das duas partes metálicas do fusível. Um fio vermelho de 50 cm, com jacaré na ponta, é levado por baixo da placa e aparece na face superior através de uma fenda, recebendo ali um nó e sendo colado. A ponta descascada do fio vermelho saindo desse ponto é soldada à outra parte metálica do fusível, constituindo-se assim o terminal + 9 a 15 Volts dc, ainda não regulados, que será levado até o painel. Observe que o fusível é mantido cerca de 1/2 cm acima da placa, sustentado pelo fio vermelho e o terminal do diodo.

OBS.
As figuras 7-1 e 7-9 encontram-se no final do livro.

PARTE B - MONTAGEM

CAPÍTULO 8
MONTAGEM

Neste capítulo é apresentado o processo de montagem do MICRO-GALENA, inclusive mapa de fiação e lista do material necessário. O que se expõe aqui é a título de sugestão, baseado na experiência adquirida na preparação do protótipo.

1 - ROTEIRO DE MONTAGEM

A montagem deve seguir um roteiro certo e planejado, pois o aparelho é frágil e delicado - como todo dispositivo eletrônico - e a execução de uma parte pode danificar outra já instalada. Aconselhamos a seqüência abaixo, que foi realizada com sucesso e sem maiores problemas:

*(1º) execução de todos os furos e aberturas necessários na placa (8 aberturas para CIs, 3 para chaves, 1 para o regulador de tensão; 10 furos para os LEDs, 4 para os parafusos dos suportes laterais, 1 para o dissipador de tensão; nos suportes laterais furar os 3 orifícios na lateral direita e 1 da ponta de prova na lateral esquerda.
Note que nenhum dos ítens acima pode ser esquecido, pois após os próximos passos a placa precisará ser manipulada com delicadeza.

*(2º) fixação da placa sobre os suportes laterais

*(3º) instalação dos 11 soquetes de CIs

*(4º) colocação das 26 chaves, colando-as à placa

*(5º) fiação e soldagem dos componentes, exceto LEDs, dissipador de calor, regulador de tensão, fiação para o terra e Vcc.

*(6º) colocação dos 11 LEDs

*(7º) instalação do dissipador de calor e regulador de tensão

*(8º) fiação para o terra e Vcc.

*(9º) colocação da ponta de prova.

Para a fonte de alimentação o único requisito é executar em primeiro lugar os furos e fendas. A seqüência restante é indiferente.

Neste ponto o MICRO-GALENA estaria pronto: bastaria encaixar os CIs nos respectivos soquetes, ligar a fonte e... nada aconteceria (fui durante 2 anos instrutor de laboratório de uma escola profissional de Eletrônica e vi algumas centenas de protótipos serem montados com o maior zelo, atenção e carinho; na hora de ligar - nada). É fatal que algum erro ou omissão tenha sido cometido durante a montagem, além da possibilidade de componentes defeituosos. No próximo capítulo veremos o processo - debugging - para localizar falhas e colocar o computador funcionando. ATENÇÃO: NÃO ENCAIXE OS CIs NOS SOQUETES APÓS A MONTAGEM; ISSO SERÁ FEITO NUMA SEQÜÊNCIA CERTA DE TESTES.

2 - PREPARAÇÃO DA PLACA

A placa, de 18 cm x 7 cm, deve ter entre 4 mm a 5 mm de espessura e ser facilmente talhável sem rachar (no protótipo usei chapa de EUCATEX; cerejeira também é adequada para o trabalho; madeiras como compensado não servem pois racham facilmente ao sofrerem um corte).

Os furos e fendas são feitos segundo a figura 7-1. Todas as posições têm uma boa margem de tolerância, não havendo preocupações milimétricas quanto a localização de soquetes, CIs e LEDs. Entretanto, seguir o mais próximo possível a figura 7-1 facilitará a aplicação do mapa de fiação.

Os cortes podem ser feitos com formão; os furos com prego ou furadeira. Em última instância contrate os serviços de um marceneiro, mas esteja certo que o trabalho é mais fácil e rápido do que possa parecer - desde que se esteja trabalhando com o tipo de madeira correto.

Na figura 7-1 as chaves são representadas no tamanho total, incluindo as duas bordas de apoio. Estas, porém, não deverão entrar na fenda, ficando apoiadas na face superior da placa. Assim, enquanto na figura 7-1 consta uma abertura de 6 cm x 2,5 cm para as chaves de dados (S7-S14), na placa o corte será de 6 cm x 1,5 cm, sobrando 0,5 cm à esquerda e 0,5 cm à direita para as bordas (essas dimensões são válidas para chave FE-AD recomendada; se usar outra marca meça a chave e altere os tamanhos se necessário). A mesma observação é válida para os blocos de chaves de endereços e controles. Se as aberturas ficarem pequenas, não cabendo as chaves, poderão ser lixadas até o tamanho ideal;

se ficarem ligeiramente maiores também não haverá problemas, pois as bordas das chaves serão coladas à placa. Em hipótese alguma as aberturas poderão igualar ou ultrapassar o tamanho total das chaves, pois não haveria como apoiá-las na placa; se tiver esse problema use massa plástica para diminuir a abertura.

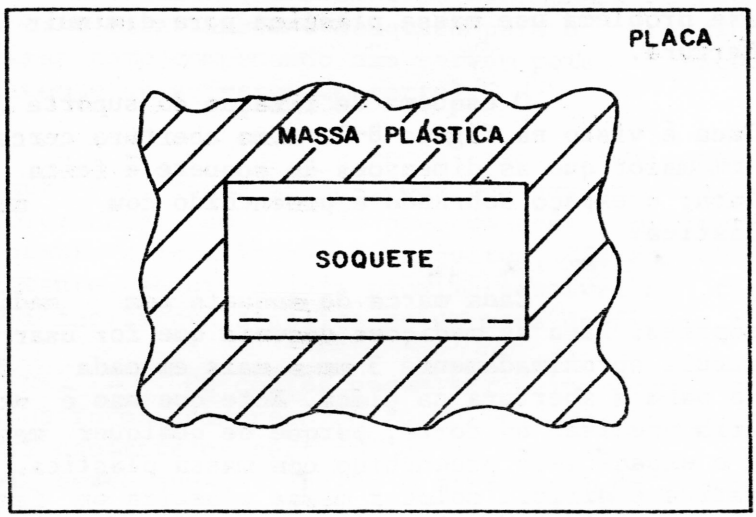
O esquema de fixação do suporte à placa é visto na figura 8-2. Uma abertura cerca de 3 mm maior que as dimensões do soquete é feita na placa; o espaço sobrando é preenchido com massa plástica.

Cada marca de soquete tem medidas próprias. Faça as medições daquela que for usar e calcule aproximadamente 3 mm a mais em cada direção para a abertura na placa. Note que não é necessário precisão no corte, porque de qualquer maneira o espaço será preenchido com massa plástica. Na prática é difícil colocar massa plástica em fendas muito estreitas, pelo que é recomendável manter essa faixa de 3 mm.

É feita uma única abertura, mais alongada, para cada par de CI 6116 (duas RAMs), 7406 (inversoras), e no canto inferior esquerdo 74LS00/74LS02. Porém, durante a fixação deve haver uma distância de 0,5 cm entre os dois soquetes, preenchida com massa plástica, garantindo espaço suficiente para as bordas dos CIs que ultrapassam ligeiramente o comprimento do soquete.

O diâmetro dos buracos dos LEDs é de 4 mm, podendo ser realizado com furadeira, prego ou parafuso. Outra opção (com muitas vantagens) é fazer um talho estreito com mais de 4 mm de comprimento, onde serão encaixados os dois terminais do

LED; neste último caso o talho deve ser feito paralelo aos lados da placa, ficando um terminal na direção da margem superior e o outro da margem inferior.



8-2 fixação do soquete à placa

A fenda para os terminais do regulador de tensão não tem maiores especificações, situando-se na casa de 1 cm x 1 cm.

Com as orientações acima e de posse das medidas dos soquetes e chaves que irá usar o leitor poderá traçar a lápis na placa de madeira o esboço de aberturas, fendas e furos a serem realizados, seguindo o esquema da figura 7-1. Não devem ser esquecidos os furos para parafusos do suporte lateral.

3 - INSTALAÇÃO DOS SOQUETES DE CIs

Um serviço todo especial - e marca característica do MICRO-GALENA juntamente com a fiação - é a instalação dos soquetes. Antes de mais todos eles, em número de 11, devem ser verificados e considerados em perfeito estado. Um soquete defeituoso pode inutilizar todo o trabalho de montagem do MICRO-GALENA, obrigando o montador a voltar à estaca zero.

A tarefa consistirá em colocar os soquetes nas respectivas aberturas da placa e preencher o espaço vazio em volta deles com massa plástica. Se um pingo desta cair nos furinhos do soquete ele ficará inutilizado, pois não haverá maneira de introduzir neles os pinos do CI, além de prováveis problemas de mau contacto. Por esse motivo cole Durex (ou similar) na face superior dos soquetes, garantindo completa vedação dos furinhos.

A massa plástica seca rapidamente, assim prepare somente a quantidade necessária a um soquete por cada vez (um catalisador é misturado com as substâncias que formarão a massa plástica, dando início à reação química cujo efeito é o endurecimento do material).

A abertura onde será instalado o soquete é totalmente coberta com Durex (ou similar), na face superior da placa. Virando-a ao contrário (portanto com a face inferior voltada para cima) teremos a parte colante do Durex livre para grudar o soquete e vedando a passagem de massa plástica pela abertura. Coloque o soquete na posição adequada, pressionando para que haja perfeita adesão ao Durex (lembre-se que o soquete está ele próprio

coberto com outro Durex). Nesta situação o soquete estará com os furinhos no mesmo nível da face superior da placa, e por baixo - face inferior - sobressairão seus pinos.

A massa plástica é agora depositada sobre o lado colante do Durex, preenchendo todo o espaço entre o soquete e as fronteiras internas da abertura. Faça o serviço cuidadosamente, evitando sujar os pinos metálicos do soquete com a massa plástica. Verifique a todo instante se a massa plástica está descolando o Durex e entrando nos furinhos do soquete. Se isto ocorrer pare tudo, limpe o soquete ou substitua-o (veja como é importante a vedação com Durex no soquete e na abertura da placa; faça-a bem feita).

Quando houver dois soquetes numa mesma abertura faça o trabalho simultaneamente para o par, como se fosse um apenas. Lembre-se de manter a distância de 0,5 cm entre eles.

Após a operação em cada abertura não retire o Durex de vedação antes da massa plástica endurecer. É melhor esperar todos os soquetes instalados para depois retirá-las.

O aspecto do CI encaixado na massa plástica (o soquete não fica visível na face superior da placa quando o CI está colocado) não chega a ser uma obra de arte. Se o montador tem pretensões quanto ao acabamento ele deve ser feito neste instante (tinta, silk-skreen ou papel colante tipo "Contact"). Máximo cuidado para não prejudicar os furinhos dos soquetes, preocupação essencial daqui para a frente.

4 - COLOCAÇÃO DAS CHAVES

Antes de colocar as chaves teste-as, uma a uma, verificando se deslizam perfeitamente nas duas direções.

A colocação é simples: passe cola (tipo Araldite) nas bordas da chave e encaixe-a na abertura correspondente da placa. Use pouca cola e tome cuidado para que ela não atinja o interior do corpo da chave (na verdade a cola é dispensável pois a própria fiação irá segurar a chave; porém a fixação prévia facilita a manipulação da placa).

5 - FIAÇÃO

A fiação é, ao lado da fixação de soquetes com massa plástica, a característica mais marcante do MICRO-GALENA. Para os veteranos de oficina será apenas uma rotina. Aos noviços, entretanto, convém observar:

-1) mantenha o ferro de solda constantemente limpo, lixando-o antes de cada soldagem.

-2) estanhe o ferro de solda e a ponta do fio a ser soldada, isto é, derreta um pouco de solda nos dois antes de executar a soldagem; se ao estanhar ficar muita solda no ferro agite-o até soltá-la.

-3) use o mínimo de solda necessário, de modo a não se formarem "bolinhos" nos pontos soldados.

Quase toda soldagem no MICRO - GALENA será entre a fiação e os pinos dos soquetes. Para

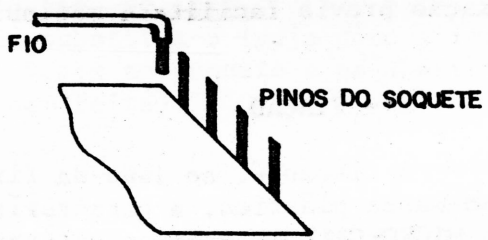
tanto use o seguinte procedimento (figura 8-3):

-1) descasque a ponta do fio e dobre-a para baixo.

-2) estanhe a ponta descascada do fio.

-3) encoste a ponta descascada do fio no pino do soquete, paralelamente (os dois na vertical) - como se fossem um único terminal.

-4) coloque o ferro de solda entre os dois e derreta um pouco de solda.



8-3 soldagem do fio ao pino do soquete

Existem algumas poucas ligações onde dois fios são soldados a um único pino. Nesse caso estanhe os fios e com o alicate de bico mantenha os dois encostados ao pino, soldando-os simultaneamente.

O mapa da fiação do MICRO-GALENA é apresentado na figura 8-4 Para não sobrecarregar o desenho foram omitidas as ligações das expansões. Veja que essa figura é a face inferior da placa do painel e corresponde exatamente ao lado oposto da figura 7-1.

As chaves são representadas como vistas do lado de baixo, sem as duas bordas. Dos LEDs constam os dois terminais e o corpo luminoso dese

nhado como traçado oval. Resistores e diodos aparecem como pequenos retângulos, e capacitores como círculos.

Os diodos e resistores ligados às chaves são colocados em posição enclinada para cima, deixando espaço para a fiação em baixo. Para soldá-los ao ponto comum Vcc (+ 5 Volts) usa-se um fio descascado comum a todos eles. Também nos pontos TERRA (GND, 0 Volt) das chaves passa-se um fio descascado pelos respectivos terminais, unindo-os.

Os 8 fios da barra de dados devem sair do microprocessador até a RAM e daí encaminhados para o latch da saída de dados; na expansão da 2a. RAM será necessário outra ligação a algum desses pontos. Para evitar sobrecarga de solda nos soquetes são usados pontos intermediários, próximos da RAM. Na figura 8- 4 eles são representados por 8 círculos cheios, acima do soquete da memória.

Com o alicate de bico molde as curvas do fio rígido tal que ele fique (quase) igual ao desenhado no mapa. Sempre que dois fios se cruzarem faça o novo passar por baixo do antigo, o que dá mais rigidez à fiação.

Em termos físicos não há preferência por qualquer ordem na execução da fiação. Porém o roteiro abaixo pode dar mais coerência e organização ao trabalho, evitando falhas e omissões:

ROTEIRO DA FIAÇÃO

-1) solde o fio descascado correspondente ao terra nas chaves de dados e endereços (ele passará pelos furos dos polos inferiores nas chaves de dados e dos polos centrais nas chaves de endere

ços).

-2) solde resistores e diodos às chaves de dados e endereços; solde o fio descascado que unirá todos ao Vcc (nas chaves de endereços unindo os anodos dos diodos; nas chaves de dados unindo os terminais livres dos resistores).

-3) ligação da barra de endereços do microprocessador às chaves (A0 até A10).

-4) ligação da barra de endereços das chaves à memória RAM (A0 até A10).

-5) ligação da barra de dados do microprocessador até os pontos intermediários; as pontas dos fios são deixadas livres para receberem a fiação da RAM e a que parte para o latch.

-6) ligação da barra de dados da RAM até os pontos intermediários; as pontas dos fios são deixadas livres.

-7) ligação dos pontos intermediários até o latch da saída de dados; todas as pontas livres são agora unidas, perfazendo 8 pontos de soldagem correspondentes a D0 até D7. Se a fiação e a soldagem foi bem feita não há perigo de contacto entre eles, colocando em curto as linhas da

---

OBS.
As figuras 7-1 e 8-4 encontram-se no final do livro.

barra de dados; em caso contrário - ou de dúvida - isole-os com fita isolante. A ponta do fio referente ao latch é deixada livre para soldagem simultânea com a fiação do buffer de entrada de dados.

-8) ligação das chaves de dados ao buffer de entrada de dados.

-9) ligação do buffer de entrada de dados até o latch; é feita soldagem simultânea com as pontas livres vindas dos pontos intermediários.

-10) instale os 8 LEDs da saída de dados; todos os anodos são unidos por um único fio descascado.

-11) solde os resistores do latch aos respectivos catodos dos LEDs da saída de dados; use os próprios terminais metálicos dos resistores, curvando-os adequadamente. É interessante fazer a soldagem dos anodos ao fio descascado num nível pouco acima das soldagens dos resistores aos catodos.

-12) faça as ligações das 6 chaves de controles mais a chave S-17 de liberação das chaves de endereços para DMA.

-13) complete as ligações que faltarem no soquete do microprocessador, exceto Vcc e terra.

-14) faça as ligações da ponta de prova lógica, exceto Vcc e terra.

-15) complete todas ligações Vcc e terra.

6 - LISTA DE MATERIAL

No projeto do MICRO-GALENA foi feito um grande esforço de seleção dos componentes, de maneira que o interessado possa encontrá-los sem maiores problemas nas lojas do ramo. Todos os CIs são largamente usados em aparelhos de consumo popular e somente não serão encontrados em caso de grave crise do mercado fornecedor.

Para facilidade do leitor apresentamos abaixo relação de lojas que dispõem dos componentes eletrônicos usados no MICRO-GALENA. No projeto tivemos ainda o cuidado de empregar material oferecido por reembolso postal, para tornar o MICRO-GALENA acessível aos leitores mais distantes e isolados. Neste caso, entretanto, o título do livro deve ser mudado para "Construa seu Computador por 1 Salário-Mínimo", já que os preços de componentes por reembolso postal são o dobro ou triplo daqueles praticados normalmente. As lojas são as seguintes, todas vizinhas, no centro do comércio eletrônico de São Paulo e geralmente com preços bastante honestos:

- MUNDISON - Comercial Eletrônica Ltda - (microprocessadores, memórias, CIs digitais e componentes em geral) - Rua Santa Ifigênia, 399.

- ELETRÔNICA RUDI LTDA - (CIs digitais e componentes em geral) - Rua Santa Ifigênia, 379.

- CINEL - Comercial Eletrônica Ltda - (CIs digitais e componentes em geral) - Rua Santa Ifigênia, 370

- TRANCHAM S/A - Indústria e Comércio - (transformadores) - Rua Santa Ifigênia, 280.

Para reembolso postal todos componentes constam da tabela da TRANCHAM.

A lista do material é acompanhada dos preços vigentes em novembro de 1985. Como o leitor pode perceber o título do livro é absolutamente correto.

LISTA DE MATERIAL

QUANTIDADE	DESCRIÇÃO/MARCA	PREÇO POR UNID.
27	chaves H-H 3 polos duplos/FE-AD ...	1.300
5	soquetes DIP 14 pinos/CELIS	3.000
1	soquete DIP 16 pinos/CELIS	3.200
2	soquetes DIP 20 pinos/CELIS	4.000
2	soquetes DIP 24 pinos/CELIS	4.200
1	soquete DIP 40 pinos/CELIS	4.500
1	CI CPU Z-80A	36.000
1	CI 6116	72.000
1	CI 74LS244	25.000
1	CI 74LS373	18.000
1	CI 74LS04	5.000
1	CI 74LS02	5.000
1	CI 74LS00	5.000
2	CI 7406	6.000
1	CI 4049B	6.000
1	CI 7805	12.000

QUANTIDADE	DESCRIÇÃO/MARCA	PREÇO POR UNID.
1	dissipador de tensão (s/especificação)	2.000
11	LED (10 vermelhos, 1 verde)	1.000
1	ponta de prova	2.500
17	diodos 1N 4004	300
1	transformador 110V-220V/9V ou ainda 110V-220V/12V), 1,5 Ampere	28.000
1	fusível 1,5 Ampere	400
1	capacitor 2.200 micro F/16 V	7.000
2	capacitor 10 micro F/10 V	1.000
1	capacitor .002 micro F	500
29	resistores 4K7 Ohms/1/8 W	200
11	resistores 470 Ohms/1/8 W	200
1	resistor 10K Ohms/1/8 W	200
1	resistor 330K Ohms/1/8 W	200
1	resistor de 100 K Ohms/1/8 W	200
2	micro-garras jacaré	2.600
1	tomada	2.000
20	metros de fio rígido número 24 encapado	400

MATERIAL NÃO INCLUÍDO NA LISTA:

massa plástica ou Durepox (pouca quantidade);
cola ARALDITE (pouca quantidade);

fita adesiva DUREX (pouca quantidade);
solda (bastante);
10 parafusos, sendo 2 com porca;
1 placa de madeira 30 cm x 18 cm x 4 cm;
1 placa de madeira 18 cm x 7 cm x 4 cm;
4 ripas de madeira (suportes laterais) - 2 de 30 cm x 3 cm x 1 cm e 2 de 18 cm x 3 cm x 1 cm;
3 metros de fio flexível (para ponta de prova e fonte de alimentação).

CAPITULO 9
DEBUGGING (Teste de Montagem)

Neste capítulo são abordadas as técnicas para testar a montagem do MICRO-GALENA. Para o leitor interessado em assistência técnica de microcomputadores deve ser observado que os procedimentos de debugging são semelhantes aos de manutenção.

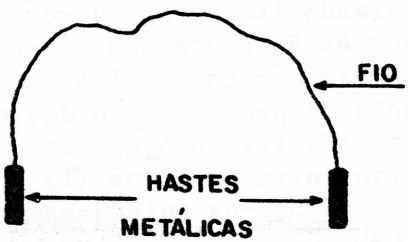
1 - DISPOSITIVOS PARA DEBUGGING

Alguns instrumentos são especialmente projetados para debugging de sistemas microprocessados. Entre eles (e adequado também para manutenção) temos o COMPUTESTER, que é descrito no livro "MANUTENÇÃO DE MICROCOMPUTADORES" do mesmo autor e editora. Se o montador dispõe de um desses aparelhos basta aplicá-lo ao MICRO-GALENA para testar a montagem e colocá-lo em funcionamento.

Para o caso mais provável de não haver qualquer dispositivo especial de debugging e, possivelmente, faltar até mesmo alguns instrumentos básicos em oficinas e laboratórios eletrônicos apresentaremos um roteiro prático de teste da montagem que os substitua,

Inicialmente devem ser providenciados 5 conectores para soquetes, isto é, fios flexí

veis que façam a ligação entre dois furinhos do soquete (como os empregados em proto-board). Cada conector terá cerca de 10 cm e nas suas duas extremidades são soldados hastes metálicas semelhantes aos pinos de CI - figura 9-1 - elas podem ser obtidas destacando-se pinos de soquetes ou de CIs estragados). Na falta dessas hastes amasse as extremidades do fio até torná-las chatas e finas; se o conector tiver pontas grossas ele alargará o furinho do soquete em que for encaixado, danificando-o.



9-1 conector de soquete

2 - ROTEIRO DE TESTE DA MONTAGEM

O roteiro para teste de montagem do MICRO-GALENA consistirá nos seguintes passos:

*1 - teste da alimentação - com todos os CIs fora dos soquetes (exceto o regulador de tensão) é feita uma inspeção visual das ligações Vcc (+5Volts) e terra. Em todos os soquetes, chaves e LEDs são verificadas as ligações Vcc e e GND, procurando-se por eventuais curtos. Se houver multímetro disponível ele é empregado neste ítem, usando-se a escala de Ohms.

*2 - teste da ponta de prova lógica - o circuito da ponta de prova lógica é examinado visualmente

(ou com o Ohmímetro) minuciosamente, garantindo-se que está correto.

*3 - teste da fonte de alimentação - o circuito da fonte de alimentação é examinado visualmente (ou com o Ohmímetro) minuciosamente, garantindo-se que está correto. Veja que os ítens (2) e (3) são fáceis, pois os circuitos são independentes e simples.

*4 - teste de ligação - o CI 4049 da ponta de prova é encaixado (todos os outros permanecem fora) os jacarés da fonte são presos aos terminais do console e a chave S1 ligada. Os dois LEDs da ponta de prova lógica devem piscar alternadamente. Se isso não ocorrer você cometeu algum erro entre os ítens (1) e (4). Desligue a fonte e refaça-os cuidadosamente até encontrar a falha.

*5 - teste da ponta de prova lógica - encostando a ponta de prova no pino 8 (terra) do CI 4049 somente o LED verde permanecerá aceso; mudando para o pino 16 (Vcc) o LED vermelho ficará aceso e o outro apagado. Caso contrário reveja todo o circuito da ponta de prova lógica.

*6 - teste da entrada de dados - encaixe todos os CIs nos soquetes, exceto o microprocessador Z-80 e a memória RAM 6116. Posicione a chave S6 em DMA (para baixo). No soquete vazio do microprocessador coloque um conector entre o pino 23 ($\overline{BUSAK}$) e o pino 24 (compartilhamento nível 1 com $\overline{WAIT}$); o LED BUSAK deve permanecer apagado. Verifique com a ponta de prova se os pinos 1 e 19 ($\overline{OEa}$ e $\overline{OEb}$) do CI 74LS244 estão em nível 1, desabilitados; ainda com a ponta de

prova confirme se a saída de dados deste CI (pinos 3,5,7,9,12,14,16,18) está em tristate; acione as chaves S-7 a S-14 nos dois sentidos, testando se a saída continua em tristate. Mude o conector do pino 24 para o 29 (terra); o LED BUSAK deve brilhar. Verifique se os pinos 1 e 19 do 74LS244 agora estão em nível 0, habilitando a entrada de dados. Acione as chaves S7 a S14 como anteriormente e para cada uma delas teste se o respectivo pino de saída tem o nível correspondente à posição da chave. Veja se o nível existente no pino de saída do 74LS 244 é o mesmo que chega ao correspondente pino de entrada do 74LS373 (3-18, 5-17, 7-14, 9-13, 12-8, 14-7, 16-4, 18-3), testando tanto nível 1 quanto nível 0.

*7 - teste da saída de dados - posicione a chave S-6 em IN (para cima). No soquete vazio do microprocessador coloque um conector entre os pinos 20 ($\overline{IORQ}$) e 37 (A7), outro entre 37(A7) e 29 (terra) e um terceiro entre 22 ($\overline{WR}$) e 24 (compartilhando nível 1 com $\overline{WAIT}$). O pino 11 (E) do CI 74LS373 deve estar desabilitado, em nível 0. Para confirmar - ou por simples prática - o sinal pode ser seguido no sentido contrário com a ponta de prova lógica: o mesmo nível 0 de desabilitação do 74LS373 será encontrado no pino 10 (saída da NOR) do 74LS02; o pino 8 (A7 na entrada da NOR) estará em 0 e o pino 9 (outra entrada da NOR) em 1; no 74LS00 o pino 8 (saída da NAND) terá nível 1, o pino 9 (entrada na NAND) apresentará nível 1 e o pino 10 (outra entrada na NAND) estará em nível 0; o CI 74LS04 mostrará nível 1 no pino 10 (saída da inversora de $\overline{IORQ}$) e nível 0 no pino 11 (entrada da inversora de $\overline{IORQ}$),

além de nível 0 no pino 8 (saída da inversora de $\overline{WR}$) e nível 1 no pino 9 (entrada da inversora de $\overline{WR}$). Teste se com o CI 74LS373 desabilitado - pino 11 em nível 0 - alterações na sua entrada não interferem na saída; para tanto mude os níveis na entrada de dados (CI 74LS244) e compare as entradas do 74LS373 com as respectivas saídas (pinos 3-2, 4-5, 7-6, 8-9, 13-12, 14-15, 17-16, 18-19). Habilite a saída de dados mudando a ligação do conector de $\overline{WR}$, colocando-o entre os pinos 22 ($\overline{WR}$) e 29 (terra). Agora o pino 11 (E-enable) do 74LS373 deve estar habilitado - em nível 1. Acompanhe o sinal de trás para a frente, como anteriormente. Nessa situação o byte na entrada do 74LS373 deve aparecer na sua saída; através das chaves S-7 a S-14 faça inicialmente o byte 0000 0000 e depois mude para 1111 1111, testando-os nos pinos da saída. Cada pino de saída do 74LS373 passa por uma inversora - coletor aberto (com 1 na entrada apresenta 0 na saída; com 0 na entrada fica com a saída em tristate). Teste com a ponta de prova lógica se as inversoras estão funcionando adequadamente (com o pino de saída do 74LS373 em nível 1 a saída da inversora será 0, permitindo a passagem de corrente de Vcc através do LED e do resistor; com o pino de saída do 74LS373 em nível 0 a saída da inversora estará em tristate, bloqueando a passagem de corrente e mantendo o LED apagado).

*8 - teste da RAM - com o MICRO-GALENA desligado o CI 6116 é encaixado no soquete (encaixe corretamente todos os pinos do CI no lado esquerdo do soquete; com um plástico - tampa de caneta esferográfica, por exemplo - pressione um a um

os pinos do lado direito até se alinharem nos respectivos furinhos do soquete; retire o CI e encaixe todos os pinos no lado direito do soquete; pressione um a um os pinos do lado aposto até se alinharem aos furinhos; certifique-se que nenhum está fora e comece a pressionar o CI para dentro do soquete, sempre observando se algum pino está entortando ou fora do furinho). Posicione as chaves S-17 em IN (para a direita), S-15 e S-16 em nível 1 (para cima) e S-6 em IN (para cima). No soquete vazio do microprocessador coloque um conector entre os pinos 21 ($\overline{RD}$) e 22 ($\overline{WR}$), entre 22 ($\overline{WR}$) e 24 (compartilhando nível 1 com $\overline{WAIT}$), entre 19 ($\overline{MREQ}$) e 1 (A11), entre 1 (A11) e 17 (compartilhando nível 1 com $\overline{NMI}$), entre 23 ($\overline{BUSAK}$) e 17 (compartilhando nível 1 com $\overline{NMI}$). Verifique com a ponta de prova se na RAM todos os pinos de endereços estão em tristate, pinos 21 ($\overline{WE}$) e 22 ($\overline{OE}$) em nível 1 e pino 18 ($\overline{CS}$) desabilitado em nível 1; em caso de erro siga o sinal de trás para a frente até localizar a origem da falha. Mude o conector que está entre os pinos 1 (A11) e 17 ($\overline{NMI}$) para 1 (A11) e 29 (terra), aterrando os dois terminais que habilitam a RAM (A11 e $\overline{MREQ}$); agora o pino 18 ($\overline{CS}$) da RAM estará em nível 0, habilitando-a. Retire os conectores dos pinos 21, 22, 19 e 1. Mude a chave S6 para DMA (para baixo). Também mude a chave S17 para DMA (para a esquerda). Ao fazer isto os níveis existentes nos pinos de endereços da RAM devem corresponder àqueles assentados nas chaves S-18 a S-28; altere uma por uma, de 1 (à esquerda) para 0 (à direita), verificando nos pinos da RAM se ocorre a mesma alteração. Além disso, os pinos 21 ($\overline{WE}$) e 20 ($\overline{OE}$) da RAM deverão ficar em nível 1 quan-

do as chaves S-15 e S-16 estiverem para cima, e em nível 0 quando forem levadas para baixo; teste as duas, individualmente. Mude o conector que está entre os pinos 23 (BUSAK) e 24 (WAIT) para 23 (BUSAK) e 29 (terra); o pino 18 ($\overline{CS}$) da RAM deve ficar em nível 0, habilitando a RAM e o LED de BUSAK acender. Deixe as chaves S-15 e S-16 em 1 (para cima), coloque o endereço 000 0000 0000 B nas chaves S-18 a S-28 e o byte 0000 0000 nas chaves S-7 a S-14; com a ponta de prova verifique se esse byte está chegando até os pinos de dados da RAM. Leve a chave S-15 ($\overline{WR}$) de 1 para 0, gravando o byte no endereço zero da RAM. Volte S-15 para cima, coloque o endereço 111 1111 1111 B nas chaves S-18 a S-29 e puxe S-15 para baixo, gravando o byte na última posição da RAM. Agora vamos fazer a leitura das duas gravações. Deixe todas as chaves S-7 a S-14 em nível 1. Coloque 000 0000 0000 nas chaves de endereços e puxe a chave S-16 ($\overline{RD}$) para baixo; com a ponta de prova verifique se o byte 0000 0000 está aparecendo nos pinos de dados da RAM. Refaça para o endereço 111 1111 1111 B testanto os dados saindo da RAM.
(ATENÇÃO: devido ao sinal existente nas chaves S-7 a S-14 durante a leitura de dados na RAM o nível 0 será percebido como um ofuscamento dos dois LEDs da ponta de prova; o nível 1 será registrado normalmente). Repita o teste gravando e lendo o byte 1111 1111 nos dois endereços.

9 - teste final do MICRO-GALENA - com a fonte desligada encaixe o CI Z-80 no seu soquete (siga as mesmas orientações vistas para o CI da RAM). As chaves de controles são posicionadas.

S4-1, S3-CK, S2-1, S16-1, S15-1, S6-IN, S17 IN, e todas as chaves S-18 a S-28 em 1. A fonte é ligada e o MICRO-GALENA estará funcionando desordenadamente, seguindo instruções aleatórias existentes na RAM. Posicione a chave S6 em DMA. O microprocessador deve responder ativando em nível 0 o pino 23 ($\overline{BUSAK}$), o que acenderá o LED de DMA. Se isso não ocorrer verifique: (1) se o pino 25 ($\overline{BUSRQ}$) está em nível 0, aterrado pela chave S6B (2); se o pino 23 ($\overline{BUSAK}$) está em nível 0, ativado pelo microprocesador. Se o pino 25 está em nível 0 mas não há ativação do pino 23, que permanece em nível 1, então algum sinal de entrada do Z-80 está errado. Comece averiguando pelo clock: encostando a ponta de prova no pino 6 (CLOCK) os dois LEDs devem brilhar simultaneamente, indício que estão piscando com alta freqüência (a mesma do clock). Os pinos 26 ($\overline{RESET}$), 17 ($\overline{NMI}$), 24 ($\overline{WAIT}$) e 16 ($\overline{INT}$) devem estar todos em nível 1. Quando o microprocessador ativar $\overline{BUSAK}$, acendendo o LED, verifique com a ponta de prova que estão em tristate todos os pinos de endereços (A0-A15), $\overline{MREQ}$, $\overline{IORQ}$, $\overline{RD}$ e $\overline{WR}$. Leve a chave S-17 para DMA e coloque o endereço 000 0000 0000 B nas chaves S18 a S28; nas chaves S-7 a S-14 coloque o byte D7=0/D6=1/D5=1/D4=1/D3=0/D2=1/D1=1/D0=0 (cuidado com a ordem, S14 é D0 e S7 é D7), correspondente à instrução "HALT" que pára o microprocessador até que ele receba um sinal negativo no pino de $\overline{RESET}$ ou $\overline{INT}$; leve a chave S15 ($\overline{WR}$) a nível 0, gravando esta instrução no endereço inicial da RAM; volte as chaves S15 para 1 e S17 para IN; leve para baixo (nível 0) a chave S2, ativando o $\overline{RESET}$; volte a chave S-6 para IN, acabando com a operação DMA. Neste ponto o compu

tador estará com a instrução "HALT" gravada no endereço zero e o microprocessador em reset. Volte a chave S2 para nível 1, desativando o reset. O Z-80 vai executar a instrução "HALT", levando seu pino 18 (HALT) a nível 0, a nível 1 (desativados) os pinos 19 (MREQ), 20 (IORQ), 21 (RD), 22 (WR). Confirme com a ponta de prova. Se tudo estiver como descrito o MICRO GALENA estará montado corretamente e pronto para ser usado.

3 - TESTE DO PASSO-A-PASSO

Acionando S3 para baixo (PP) o MICRO GALENA passa a funcionar no sistema passo-a-passo, comandado pela chave S4. Dois testes interessantes podem ser realizados:

(a) RESTART A PASSO-A-PASSO: selecionado o passo-a passo verifique com a ponta de prova os pinos do Z-80, especialmente os de endereços. Acione o restart (chave S2 para baixo), mantendo a ponta de prova num dos pinos de endereço e em seguida acionando várias vezes a chave passo - a passo (S4). O pino de endereço mudará para tristate, sinal que o Z-80 entrou no ciclo de restart. Volte S2 para cima e acione S4 até aparecer o endereço 0000 0000 0000 0000B nos pinos de Z-80.

(b) DMA PASSO-A-PASSO: selecionado o passo-a-passo acione S6 para baixo, solicitando DMA. Dê vários pulsos com S4 até o LED de BUSAK acender; neste instante teste com a ponta de prova os pinos do Z-80, confirmando que estão em tritaste os terminais de endereços e controles MREQ,

$\overline{IORQ}$, $\overline{RD}$, $\overline{WR}$.

4 - TESTE DE MAU CONTACTO

Mau contacto é a falha mais comum não só no MICRO-GALENA como em microcomputadores e equipamentos digitais.

Observe que um mesmo sinal está presente em vários pontos do circuito. Por exemplo, A7 está no pino 1 da RAM 6116, pino 37 do Z-80, e pino 8 da 74LS02. Coloque o computador em operação e verifique com a ponta de prova nos respectivos pinos dos CIs se o sinal é o mesmo; o teste deve ser feito no pino do CI e não no pino do soquete. Se há diferença algo está errado. Em especial, se um deles está em tristate e os outros não trata-se de mau contacto no soquete.

5 - CALIBRAÇÃO DO CLOCK

Se todo o circuito está correto mas o computador não funciona adequadamente uma causa provável é excesso de freqüência do clock.

Inicialmente coloque a chave S3 para cima (CK) e confirme no pino 6 do Z-80 se ele está recebendo o sinal (os dois LEDs da ponta de prova devem ficar acesos simultaneamente; na verdade estão piscando a velocidade muito alta). Puxe S3 para baixo (PP) e acione S4 para cima e para baixo, verificando se os dois níveis são recebidos no pino 6 do Z-80.

Substitua o resistor de 150 ohms do oscilador do clock (entre os pinos 3 e 1 do CI 74LS04) por outro de 1KOhms. Se o MICRO-GALENA ago

ra funciona normalmente trata-se realmente de excesso na freqüência do clock. Substitua o resistor de 1 KOhms por outro de valor menor (470 Ohms), e teste novamente o funcionamento. Faça substituições para resistências menores até atingir o menor valor, que corresponderá à máxima freqüência.

PARTE C - PROGRAMAÇÃO

CAPÍTULO 10
IMPLEMENTAÇÃO DO ASSEMBLER/CÓDIGO DE MÁQUINAS

Iniciaremos este capítulo demonstrando como se grava e se lê diretamente na memória (DMA) do MICRO-GALENA. A seguir são discutidas as instruções do Z-80, com exemplos de sua implementação no MICRO-GALENA. A lista de instruções do Z-80 é encontrada no Apêndice A, no final do livro.

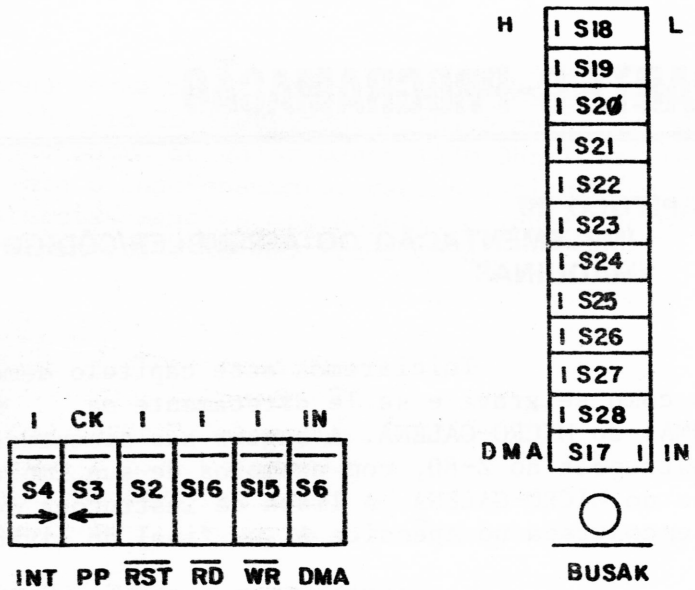
1 - PROGRAMANDO O MICRO-GALENA

Para entrar com um programa na memória do MICRO-GALENA utilize o seguinte roteiro:

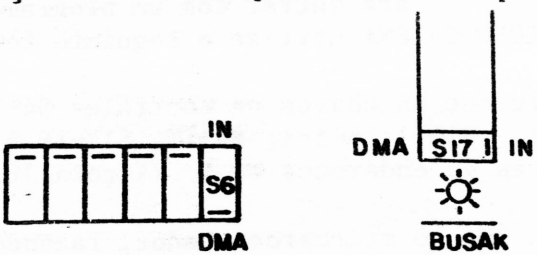
*1 - posicione as chaves de controles S4-1, S3 - CK, S2-1, S16-1, S15-1, S6-IN, S17-IN e todas as chaves de endereços em 1. (figura 10-1).

*2 - peça DMA ao microprocessador, fazendo S6 - DMA (para baixo). Quando o LED BUSAK acender libere as chaves de endereços posicionando S17 em DMA (figura 10-2).Se o microprocessador não ativar $\overline{\text{BUSAK}}$ deixe a chave S6 em DMA e acione o reset (S2 para baixo); ao sair do reset (S2 para cima) $\overline{\text{BUSAK}}$ será ativado.

*3 - posicione as chaves S18 a S28 no endereço zero (todas para a direita); nas chaves S7 - S14 coloque o byte referente à 1a. instrução do programa; puxe a chave S15-$\overline{\text{WR}}$ para baixo e reposicione-a para cima;



10-1 posição das chaves para início da programação

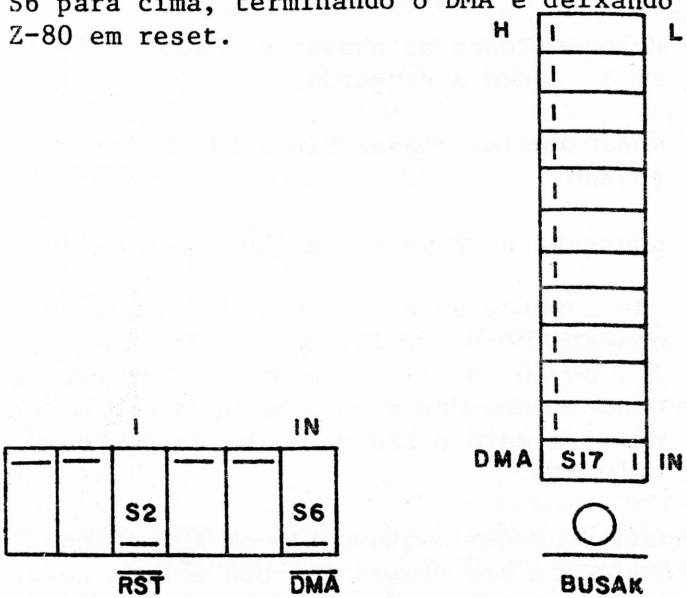


10-2 posição das chaves após o LED BUSAK acender

*4 - posicione as chaves S18 a S28 no endereço 1 (S18 para a esquerda e todas as outras na direita); coloque o próximo byte de instrução nas chaves S7-S14 e acione S15 para baixo, voltando-a depois para cima. Repita a operação para todos os bytes de instrução, cada qual em seu endereço.

*5 - volte a colocar o byte do endereço zero nas chaves S7-S14.

*6 - posicione todas as chaves de endereço em 1 (para a esquerda) e desative-as levando S-17 para IN; se necessário altere agora as chaves S7-S14; continue com S6 em DMA e faça reset (figura 10-3), o que apagará o LED BUSAK. Puxe S6 para cima, terminando o DMA e deixando o Z-80 em reset.



10-3 preparação das chaves para finalizar o DMA

*7 - puxando S2 para cima terminará o reset e o Z-80 irá procurar no endereço zero a primeira instrução a executar.

Devido ao seu sistema de chaveamento o MICRO-GALENA escreve no endereço zero o byte que estiver nas chaves S7-S14 quando a chave S17 é retornada de DMA para IN; por isso foi incluído o item 5 acima.

2 - LEITURA DE MEMÓRIA

A leitura dos dados gravados na memória - para conferir o programa ou outra finalidade- segue o roteiro abaixo:

*1 - peça DMA e libere as chaves de endereços como nos itens (1) e (2) da seção anterior (figuras 10-1 e 10-2).

*2 - coloque todas as chaves de dados - S7 a S14 - em 1 (para a esquerda).

*3 - posicione as chaves S18 a S28 no endereço desejado.

*4 - mantenha a chave S16 em $\overline{RD}$ (para baixo)

*5 - com a ponta de prova teste os pinos de dados da RAM (D0-0, D1-10, D2-11, D3-13, D4-14, D5-15, D6-16, D7-17). O nível 0 aparecerá como um ofuscamento dos dois LEDs da ponta de prova; o nível 1 será o LED vermelho aceso, como normalmente.

*6 - continue com a chave S16 em $\overline{RD}$, coloque novo endereço nas chaves S18-S28 e faça nova leitura; ao terminar deixe S16 em 1 (para cima).

*7 - para finalizar a leitura assente nas chaves S7-S14 o byte do endereço zero, puxe S17 para IN e todas S18-S28 para 1, altere S17-S14 se necessário e retire S6 de DMA para IN.

3 - ENTRADA DMA SIMPLES E SAÍDA DE DADOS

A primeira rotina a ser apresentada

para o MICRO-GALENA será uma entrada DMA simples, para gravação de poucos bytes na memória (veremos adiante um processo mais sofisticado para gravações maiores). Acrescentando com uma rotina de saída de dados (mostrando-os nos 8 LEDs) teremos o MICRO GALENA em condições de realizar as duas extremidades do processo de computação (entrada de dados - processamento - saída de dados).

A instrução "IN A, (n)" copia no acumulador o byte existente na porta de número n, que pode ser de zero a 255. No MICRO-GALENA não foi atribuída numeração à porta de entrada de dados, de maneira que podemos usar qualquer número no lugar de n. O formato desta instrução é

ASSEMBLER	CÓDIGO DE MÁQUINA
IN A, (n)	1101 1011/.....

Os traços indicam o espaço de 1 byte que deve ser preenchido com o número da porta, em binário. Note que esta instrução ocupará 2 byte na memória.

Para saída de dados usamos "OUT (n), A", que copia o conteúdo do acumulador na porta de número n. No MICRO-GALENA os 8 LEDs da saída de dados são habilitados quando A7=0, não importando os níveis de A0 até A6. Portanto, n será o byte 0XXX XXXX, onde X representa 1 ou 0 escolhidos arbitrariamente. O formato da instrução é

ASSEMBLER	CÓDIGO DE MÁQUINA
OUT (n),A	1101 0011/.....

Também ocupará 2 bytes na memória.

Ao sair de reset o Z-80 executa as instruções gravadas na memória, começando no endereço zero. Se não houver uma instrução de parada no fim do programa ele continuará executando falsas instruções existentes aleatoriamente nos endereços restantes, com conseqüências imprevisíveis. Usamos a instrução "HALT", que mantém indefinidamente o mesmo endereço no registro PC (contador de programa) até que seja recebido um sinal de reset ou interrupção. O formato da instrução é

ASSEMBLER	CÓDIGO DE MÁQUINA
HALT	0111 0110

Ao executá-la o Z-80 ativa o pino 18 (nível 0).

Nosso primeiro programa está pronto. Consiste na entrada de 1 byte de dados pelas chaves S7-S14, que será apresentado nos 8 LEDs de saída.

PROGRAMA 1 - Entrada e saída de 1 byte de dados:

ASSEMBLER

```
LABEL  MNEM.  OPERANDO      COMENTÁRIOS
       ORG    0            ; endereço inicial do progra-
                           ; ma
       IN     A,(255)      ; entrada de 1 byte pela por
                           ; ta 1111 1111B
       OUT    (127),A      ; saída do byte pela porta
                           ; 0111 1111B
       HALT                ; fim do programa
```

A primeira linha ("ORG 0") é uma pseudo instrução desconhecida pelo microprocessador mas que indicará ao tradutor para o código de máquina onde deve alocar na memória a primeira instrução do programa (endereço zero). Os comentários também são desconhecidos pelo microprocessador, servindo apenas de esclarecimento para o leitor do programa e sempre antecedidos de (;). Note que no assembler usamos numeração decimal

PROGRAMA 1

CÓDIGO DE MÁQUINA

ENDEREÇO NA MEMÓRIA	CÓDIGO GRAVADO	(ASSEMBLER CORRESPONDENTE)
0000 0000 0000 0000	1101 1011	IN A (n)
0000 0000 0000 0001	1111 1111	
0000 0000 0000 0010	1101 0011	OUT (n), A
0000 0000 0000 0011	0111 1111	
0000 0000 0000 0100	0111 0110	HALT

Para implementar o programa no MICRO-GALENA siga o roteiro da seção 1 deste capítulo. Eventualmente também faça a leitura do programa após gravá-lo, como descrito na seção 2.

Coloque um byte qualquer nas chaves S7-S14 e saia do reset (S2 para cima). O MICRO-GALENA executará o programa, lendo o byte e escrevendo o nos 8 LEDs. Volte ao reset (S2 para baixo) e coloque outro byte. Ao sair do reset ele será escrito nos LEDs.

Após o byte aparecer nos LEDs verifique com a ponta de prova o pino 18 (HALT) do Z-80. Ele estará em nível 0, indicando que o microproces-

sador executou a instrução "HALT".

4 - TRANSFERÊNCIA DE DADOS ENTRE REGISTROS

A instrução "LD r, n" carrega o byte n em qualquer dos registros A,B,C,D,E,H,L. Ao escrever o programa em assembler a letra r é substituída pelo registro. Em código de máquina a instrução é codificada "00 .... 110/.....". Nos três espaços vagos do início acrescenta-se um código de três bits correspondente ao registro desejado: B=000, C=001, D=010, E=011, H=100, L=101, A=111; se for acrescentado 110 o byte será transferido para a posição de memória cujo endereço é o conteúdo do par HL, o que é representado por (HL).

No programa 2 abaixo os registros H e L são carregados com 1 byte cada um, formando um endereço de 2 bytes = 16 bits que passa a ser indicado por '(HL)'. Um terceiro byte é então carregado na posição de memória indicada por (HL).

PROGRAMA 2 - carga de um byte na posição indicada pelo ponteiro (HL).

ASSEMBLER

```
LABEL   MNEM.   OPERANDO      COMENTÁRIOS
        ORG       0       ; locação da 1a. instrução do
                            programa
        LD       H,1      ; carrega 0000 0001B em H
        LD       L,128    ; carrega 1000 0000B em L
        LD       (HL),255 ; carrega 1111 1111B no ende
                          ; reço 0000 0001 1000 0000B =
                          ; 384
        HALT              ; fim do programa
```

PROGRAMA 2

CÓDIGO DE MÁQUINA

ENDEREÇO NA MEMÓRIA	CÓDIGO	(ASSEMBLER)
0000 0000 0000 0000	0010 0110	LD H,1
	0001 0000 0001	
	0010 0010 1110	LD L,128
	0011 1000 0000	
	0100 0011 0110	LD (HL),255
	0101 1111 1111	
	0110 0111 0110	HALT

Grave este programa na memória. Ao sair de reset e executá-lo o Z-80 escreverá o byte 1111 1111 no endereço 384. Com a ponta de prova confirme. A seguir reescreva na posição 0101B=5 um novo byte (pode ser 0000 0000), mantendo o resto do programa. Após o reset e a execução verifique se o novo byte está gravado no endereço 384. Reescreva novos bytes nas posições 0001B=1 e 0011B=3, alterando o endereço onde ocorrerá a gravação (mas sem ultrapassar o limite de 2K da memória).

A instrução "LD A, r" carrega no acumulador A o conteúdo de um dos registros B,C,D,E,H, L ou da posição de memória indicada por (HL). O seu código é "0111 1...", preenchendo-se os espaços

vazios com os três bits que determinam qual dos registros será operado (B=000, C=001, D=010, E=011, H=100, L=101, (HL)=110, A=111).

Inversamente "LD r,A" carrega no registro r o conteúdo existente no acumulador A. Seu código é "01....111", completado pelos três do registro.

Nessas duas últimas instruções e nas outras de transferência de dados entre registros o byte é copiado no destino sem ser alterado na fonte. Por exemplo, se fizermos "LD A,B" o conteúdo de B será copiado no acumulador, mas após a execução da instrução permanecerá inalterado no registro B.

Algumas páginas atrás recomendamos ao leitor que conferisse o programa gravado na memória usando o DMA e a ponta de prova. O programa 3 substitui essa tarefa, lendo os bytes gravados na memória e apresentando-os nos 8 LEDs.

O byte 0000 0000 é carregado no registro H, formando a parte superior do endereço. O byte inferior do endereço é gravado no acumulador através de entrada de dados pela porta; do acumulador ele é transferido para o registro L. O par HL agora é um ponteiro cujo conteúdo aponta o endereço desejado na memória. O byte de instrução existente ali é transferido para o acumulador (mas sem ser alterado na fonte) e dali apresentado nos LEDs através de saída de dados por porta.

PROGRAMA 3 - Leitura de bytes gravados na memória

ASSEMBLER

```
LABEL  MNEM.   OPERANDO      COMENTÁRIOS
       ORG       0          ; endereço da 1a. instrução
       LD        H,0        ; carrega 0000 0000B no regis
                            ; tro H
       IN        A,(255)    ; entra com algum byte pela
                            ; porta 1111 1111B
       LD        L,A        ; byte recebido é carregado no
                            ; registro L, formando a parte
                            ; inferior do endereço
       LD        A,(HL)     ; byte gravado no endereço in-
                            ; dicado por (HL) é copiado no
                            ; acumulador
       OUT       (127),A    ; byte copiado no acumulador
                            ; sái pela porta 0111 1111B
       HALT                 ; fim do programa
```

PROGRAMA 3

CÓDIGO DE MÁQUINA

ENDEREÇO NA MEMÓRIA	CÓDIGO	(ASSEMBLER)
0000 0000 0000 0000	0010 0110	LD H,0
0001	0000 0000	
0010	1101 1011	IN A, (255)
0011	1111 1111	
0100	0110 1111	- LD L,A
0101	0111 1110	- LD A,(HL)
0110	1101 0011	OUT (127),A
0111	0111 1111	
1000	0111 0110	HALT

Após a gravação e enquanto o Z-80 estiver em reset coloque nas chaves de dados o byte 0000 0000. Ao tirá-lo do reset este byte dá entrada no acumulador e o código "0010 0110" da 1a. instrução será escrito nos 8 LEDs. Faça novamente reset e assente o byte 0000 0001 nas chaves de dados; ao sair do reset o 2º byte gravado na memória (0000 0000) será apresentado nos LEDs. Repita a operação para os 8 endereços onde o programa está gravado.

Utilizamos o programa para conferir os códigos de instruções gravados, mas ele é válido para leitura de qualquer posição da memória.

Apresentamos as instruções "LD A,r" e "LD r, A". Na verdade qualquer registro pode substituir o acumulador no operando. As instruções ficam neste caso geral como "LD r1, r2", com o código de máquina "01 DD DSSS", onde DDD são os três bits que caracterizam o registro de destino r1 e SSS os três do registro fonte r2. Veja que preen

chendo DDD ou SSS pelos bits 111 correspondentes ao acumulador recairemos nos códigos usados para "LD A,r" e "LD r,A".

5 - TRANSFERÊNCIA DE DADOS ENTRE PARES DE REGISTROS

A instrução "LD rr, nn" carrega os dois bytes nn num dos pares de registros BC, DE,HL, SP. Seu código é 00..0001/.... ..../.... ....,preenchendo-se os 16 bits finais com os dois bytes nn e os dois bits no começo conforme a correspondência: BC=00, DE=01, HL=10, SP=11.

O programa seguinte é semelhante ao anterior, porém o endereço a ser lido na memória é carregado diretamente no par HL. O conteúdo deste endereço é copiado no acumulador e sai pelos 8 LEDs.

PROGRAMA 4 - Leitura do conteúdo de um endereço

ASSEMBLER

```
LABEL   MNEM.   OPERANDO        COMENTÁRIOS
        ORG     0           ; endereço da 1a. instrução
        LD      HL,0        ; carrega 0000 0000/ 0000
                            ; 0000B em HL
        LD      A,(HL)      ; conteúdo do endereço indi-
                            ; cado por (HL) é copiado no
                            ; acumulador
        OUT     (127),A     ; byte copiado no acumulador
                            ; sai pela porta 0111 1111B
        HALT                ; fim do programa
```

PROGRAMA 4

CÓDIGO DE MÁQUINA

ENDEREÇO NA MEMÓRIA	CÓDIGO	(ASSEMBLER)
0000 0000 0000 0000	0010 0001	LD HL,0
0001	0000 0000	
0010	0000 0000	
0011	0111 1110	- LD A,(HL)
0100	1101 0011	OUT (127),A
0101	0111 1111	
0110	0111 0110	- HALT

Ao sair do reset o Z-80 escreverá "0010 0001" nos 8 LEDs, que é o byte gravado no endereço 0. Para fazer a leitura do conteúdo de outro endereço basta alterar o 2º e 3º bytes do programa (no assembler mudando o zero na instrução "LD HL, 0").

O acumulador copia seu conteúdo no endereço indicador por (BC), (DE) ou (nn) através da instrução "LD (rr), A". nn representa 2 bytes escolhidos pelo programador. O código é 00..0010, preenchendo os dois bits com BC=00, DE=01 e nn= 11. Para este último acrescenta-se mais 2 bytes no final da instrução (0011 0010/.... ..../.... ....), sendo que o 2º byte determinará os 8 bits inferio

res do endereço e o 3º os 8 bits superiores (por exemplo: 0011 0010/0000 0001/0000 0000 copiará o conteúdo do acumulador no endereço 0000 0000/ 0000 0001). Esta inversão de bytes superior e inferior ocorrerá em todas instruções com endereço explícito.

A instrução "LD A, (rr)" é o inverso da que acabamos de ver, copiando no acumulador o conteúdo do par rr, que pode ser BC, DE ou nn. O código é 00..1010, com a correspondência dos dois bits a serem preenchidos e a observação sobre nn como na anterior.

Observe que as instruções "LD (HL), A" e "LD A, (HL)" não se enquadram acima, pois sendo um ponteiro especial (HL) pode ser usado na instrução "LD r1, r2", estudada na seção 2 deste capítulo.

O programa seguinte é semelhante ao programa 4, apenas usando o par BC ao invés de HL. O endereço zero é carregado em BC, o conteúdo deste endereço - que é o byte da 1a. instrução - copiado no acumulador e gravado nos 8 LEDs.

PROGRAMA 5 - Leitura do conteúdo de endereço usando par BC

ASSEMBLER

```
LABEL  MNEM.  OPERANDO      COMENTÁRIOS
       ORG      0          ; endereço da 1a. instrução
       LD       BC,0       ; carrega endereço 0000 0000/
                           ; 0000 0000B no par BC
       LD       A,(BC)     ; conteúdo do endereço carre
                           ; gado em BC é copiado no
                           ; acumulador
       OUT      (127),A    ; conteúdo do acumulador sái
                           ; pela porta 0111 1111B
       HALT                ; fim do programa
```

PROGRAMA 5

CÓDIGO DE MÁQUINA

ENDEREÇO NA MEMÓRIA	CÓDIGO	(ASSEMBLER)
0000 0000 0000 0000	0000 0001	LD BC,0
0001	0000 0000	
0010	0000 0000	- LD A,(BC)
0011	0000 1010	
0100	1101 0011	OUT (127),A
0101	0111 1111	
0110	0111 0110	- HALT

Ao sair do reset o Z-80 escreverá nos LEDs "0000 0001", que é o byte gravado no endereço zero.

Outro programa equivalente ao programa 4 é apresentado a seguir, usando (nn) no lugar dos pares de registros. O conteúdo do endereço 4=0000 0000/0000 0100B é copiado no acumulador e daí sái para os 8 LEDs. No código de máquina observe a posição do byte superior (0000 0000) e do inferior (0000 0100) do endereço.

PROGRAMA 6 - Leitura do conteúdo de um endereço imediato

ASSEMBLER

```
LABEL  MNEM.   OPERANDO      COMENTÁRIOS

       ORG      0          ; endereço da 1a. instrução
       LD       A,(4)      ; copia no acumulador conteúdo
                           ; do endereço 0000 0000/ 0000
                           ; 0100B
       OUT      (127),A    ; conteúdo copiado sái pela
                           ; porta 0111 1111B
       HALT                ; fim do programa
```

PROGRAMA 6

CÓDIGO DE MÁQUINA

ENDEREÇO NA MEMÓRIA	CÓDIGO	(ASSEMBLER)
0000 0000 0000 0000	0011 1010	LD A, (4)
0001	0000 0100	
0010	0000 0000	OUT (127),A
0011	1101 0011	
0100	0111 1111	
0101	0111 0110	- HALT

Após o reset o Z-80 gravará nos 8 LEDs "0111 1111", que é o byte gravado no endereço 4.

Com "LD (nn), rr" ocorre uma interação entre o par BC ou DE e os dois bytes dos endereços nn e nn+1; o byte contido em C ou E é copiado em nn e o outro byte conteúdo de B ou D copiado em nn+1. O código da instrução é 1110 1101/01..0011 / ..../.... .... . No sentido inverso "LD rr, (nn)", código 1110 1101/01.. 1011/.... ..../ .... copia no registro C do par BC ou E do par DE o conteúdo do endereço nn e em B ou D o conteúdo de nn+1.

Os conteúdos de nn e nn+1 também interagem com o par HL através das instruções "LD HL, (nn)", código 0010 1010/.... ..../.... .... (o conteúdo de L é copiado em nn e o de H em nn+1).

Comparando "LD rr, (nn)" com "LD HL, (nn)", e suas inversas "LD (nn), rr" - "LD (nn), HL", observamos que fazem a mesma operação, porém as instruções com o par HL gastam menos espaço de memória - 3 bytes contra 4 bytes dos outros registros.

No programa 7 o registro D é carregado com um byte e E com outro; o conteúdo do par DE é copiado numa posição de memória.

PROGRAMA 7 - Copia na memória o conteúdo de um par de registro

ASSEMBLER

```
LABEL  MNEM.  OPERANDO      COMENTÁRIOS
       ORG      0          ; endereço da 1a. instrução
       LD       D,0        ; carrega byte 0000 0000 em  D
       LD       E, 255     ; carrega byte 1111 1111 em  E
       LD       (16),DE    ; copia conteúdo de DE em 0001
                           ; 0000B
       HALT                ; fim do programa
```

PROGRAMA 7

CÓDIGO DE MÁQUINA

ENDEREÇO NA MEMÓRIA	CÓDIGO	(ASSEMBLER)
0000 0000 0000 0000	0001 0110	LD D,0
0001	0000 0000	
0010	0001 1110	LD E,255
0011	1111 1111	
0100	1110 1101	
0101	0101 0011	LD (16),DE
0110	0000 0000	
0111	0001 0000	
1000	0111 0110	- HALT

Após o reset pede-se novamente DMA e é feita a leitura da memória com a ponta de prova: no endereço 16 (0001 0000B) será encontrado o byte 1111 1111 carregado em E, no endereço 17 (0001 0001B) estará registrado 0000 0000 correspondente a D.

6 - ENTRADA E SAÍDA SELECIONADA PELO REGISTRO C

As instruções de entrada e saída que vimos são centradas no acumulador - IN A, (n) e OUT (n), A. Mas com "IN r, (C)", código 1110 1101/01.. .000, a entrada indicada pelo conteúdo de C é copiada no registro. A instrução inversa "OUT (C), r", código 1110 1101/01.. .001, coloca na saída indicada por C o conteúdo do registro r ("IN r, (C)" altera as flags, como veremos adiante; (HL) - código 110 - não pode ser usado nestas instruções).

No programa 8 o registro C indica a porta por onde entra um byte para L (note que qualquer conteúdo de C é válido, pois a porta de entrada do MICRO-GALENA não é numerada). O par HL passa a apontar para um endereço na memória, cujo conteúdo é copiado no registro H. O byte 0111 1111 é carregado em C para indicar a porta de saída dos 8 LEDs e o conteúdo de H sai por ela.

PROGRAMA 8 - Leitura e apresentação do conteúdo de um endereço programável

ASSEMBLER

```
LABEL  MNEM.  OPERANDO      COMENTÁRIOS
       ORG    0            ; endereço da primeira instru-
                           ; ção
       LD     H,0          ; carrega 0000 0000B em H
       IN     L,(C)        ; entra com byte em L
       LD     H,(HL)       ; copia conteúdo do endereço
                           ; indicado por (HL) em H
       LD     C,127        ; carrega número da porta de
                           ; saída em C
       OUT    (C),H        ; conteúdo de H é posto na
                           ; porta de saída
       HALT                ; fim do programa
```

PROGRAMA 8

CÓDIGO DE MÁQUINA

ENDEREÇO NA MEMÓRIA	CÓDIGO	(ASSEMBLER)
0000 0000 0000 0000	0010 0110	LD H,0
0001	0000 0000	
0010	1110 1101	IN L,(C)
0011	0110 1000	
0100	0110 0110	LD H,(HL)
0101	0000 1110	
0110	0111 1111	OUT (C),H
0111	1110 1101	
1000	0110 0001	HALT
1001	0111 0110	

Enquanto o Z-80 estiver no reset coloque o byte 0000 0000 nas chaves de dados. Ao sair do reset ele entrará com esse byte no registro L e apresentará "0010 0110" nos 8 LEDs, correspondente ao conteúdo do endereço 0000 0000/0000 0000B. Faça novamente reset e ponha "0000 0001" nas chaves de dados; nos LEDs aparecerá "0000 0000", referente ao byte gravado no endereço 1. Repita para os 10 endereços do programa e confira.

7 - CARGA E TRANSFERÊNCIA COM OS REGISTROS IX-IY

Os registros de índice IX e IY, de 16 bits, copiam os conteúdos dos endereços nn e n+1 através da instrução "LD rr, (nn)", código 11-1 1101/0010 1010/.... ..../.... ...., preenchendo-se o bit vago no início com 0 para IX e 1 para IY; o conteúdo de nn é copiado nos 8 bits inferiores e de nn+1 nos 8 bits superiores. A instrução inversa é "LD (nn), rr", código 11-1 1101/0010 0010/ ..../.... ...., que copia o conteúdo de 16 bits de um dos registros nos endereços nn e nn+1. Veja que o assembler é semelhante ao das instruções para os pares BC e DE, mas o código de máquina é diferente; o par HL também tem o mesmo assembler e código de máquina de apenas 3 bytes.

2 bytes podem ser diretamente carregados nestes registros de índice pela instrução "LD rr, nn", código 11-1 1101/0010 0001/.... ..../.... ...., com IX=0 e IY=1 no bit inicial vago; nn é copiado no byte inferior do registro e nn+1 no byte superior. Como anteriormente o assembler é igual ao de carga para os outros pares, mas o código de máquina é diferente.

A notação '(IX+d)' ou '(IY+d)' significa o byte indicado pelo endereço obtido somando-se o conteúdo do IX ou IY com d. Por exemplo, se o conteúdo de IX é 0000 0000 0000 0001B=1 e d=2=10B a notação (IX+2) indicará o conteúdo do endereço 0000 0000 0000 0011B=3.

Com a instrução "LD r, (rr+d)" o conteúdo do endereço (rr+d) é copiado no registro r; seu código é 11-1 1101/01.. .110/dddd dddd, preenchendo-se o bit vago no início com 0 para IX ou 1 para IY e os três espaços com a correspondência de registros já apresentada.

No programa 9 IX é carregado com o endereço seis, o acumulador copia o conteúdo de (IX+3) - portanto o conteúdo do endereço 9 - e o coloca na porta de saída.

PROGRAMA 9 - Apresentação de byte apontado por registro de índice deslocado

ASSEMBLER

```
LABEL  MNEM.  OPERANDO      COMENTÁRIOS
       ORG    0            ; endereço da 1a. instrução
       LD     IX,6         ; carrega 0000 0000/0000 0110
                           ; em IX
       LD     A,(IX+3)     ; copia conteúdo de (IX+3) no
                           ; acumulador
       OUT    (127),A      ; coloca conteúdo do acumula
                           ; dor na porta de saída
       HALT                ; fim do programa
```

PROGRAMA 9

CÓDIGO DE MÁQUINA

ENDEREÇO NA MEMÓRIA	CÓDIGO	(ASSEMBLER)
0000 0000 0000 0000	1101 1101	LD IX,6
0001	0010 0001	
0010	0000 0110	
0011	0000 0000	
0100	1101 1101	LD A,(IX+3)
0101	0111 1110	
0110	0000 0011	
0111	1101 0011	OUT (127),A
1000	0111 1111	
1001	0111 0110	- HALT

Ao sair do reset o Z-80 escreverá "0111 0110" nos 8 LEDs, correspondente ao byte gravado no endereço 9. Alterando o valor do deslocamento gravado no endereço 7 obtem-se o byte existente em qualquer posição de 6 a 261.

A instrução "LD (rr+d), n", código 11-1 1101/0011 0110/dddd dddd/.... ...., carrega em (IX+d) ou (IY+d) o byte imediato n. Igualmente "LD (rr+d),r", código 11-1 1101/01.. .110/dddd dddd, copia em (IX+d) ou (IY+d) o conteúdo de um registro selecionado pelos três bits do 2º byte. Nestas duas últimas instruções d é um número em complemento 2 (veja Capítulo 1), o que permite deslocamentos para frente (positivo) ou para trás (negativo). Portanto, o deslocamento ficará entre 0111 1111B=+127 e 1000 0001B=-127.

No programa 10 o registro IY é carregado com 16; um byte entra pela porta (qualquer

uma) apontada por C e é gravado em B, sendo daí copiado no endereço (IY-1) - ou seja, endereço 15; o acumulador é carregado com o conteúdo do endereço 15 e colocado na porta de saída. O programa termina no endereço 14, de modo que a gravação de dados na posição 15 fica fora dele e não o altera.

PROGRAMA 10 - Cópia de byte em endereço indicado por IY deslocado

ASSEMBLER

```
LABEL   MNEM.   OPERANDO      COMENTÁRIOS
        ORG     0           ; endereço da instrução ini
                            ; cial
        LD      B,(C)       ; entra com byte em B
        LD      IY,16       ; carrega 0000 0000 0001 0000
                            ; B em IY
        LD      (IY-1),B    ; copia B em (IY-1) = endere
                            ; ço 15
        LD      A,(15)      ; copia conteúdo do endereço
                            ; 15 no acumulador
        OUT     (127),A     ; grava conteúdo do acumula
                            ; dor na porta de saída
        HALT                ; fim do programa
```

PROGRAMA 10

CÓDIGO DE MÁQUINA

ENDEREÇO NA MEMÓRIA	CÓDIGO	(ASSEMBLER)
0000 0000 0000 0000	1110 1101	IN B,(C)
0001	0100 0000	
0010	1111 1101	
0011	0010 0001	LD IY,16
0100	0001 0000	
0101	0000 0000	
0110	1111 1101	
0111	0111 0000	LD(IY-1),B
1000	1111 1111	
1001	0011 1010	
1010	0000 1111	LD A,(15)
1011	0000 0000	
1100	1101 0011	OUT(127),A
1101	0111 1111	
1110	0111 0110	- HALT

Durante o reset coloque um byte na chave de dados. Ao sair do reset o Z-80 o gravará nos 8 LEDs. Volte ao reset e repita para outro byte.

Note no código de máquina como os dois bytes do número 16 na instrução "LD IY,16" são escritos na ordem inversa (o mesmo acontece em "LD A,(15)"). Veja ainda o uso do complemento 2 na instrução "LD (IY-1), B".

8 - CARGA E TRANSFERÊNCIA COM O PONTEIRO DE PILHA - SP

O SP ("stack pointer") segue as mesmas instruções vistas para os outros registros de 16 bits - LD SP, nn/LD (nn), SP - acrescentando-se os bits 11 para caracterizá-lo.

O SP pode copiar o conteúdo existente no par HL pela instrução "LD SP, HL", código 1111 1001. O mesmo ocorre para os conteúdos dos registros de índice - "LD SP,rr", código 11-1 1101/1111 1001, preenchendo-se o bit vago com 0 para IX e 1 para IY.

As duas instruções características do SP são PUSH (ponha) e POP (retire). Com "PUSH rr", código 11.. 0101, o conteúdo de um dos pares de registros (BC=00, DE=01, HL=10, AF=11) é posto no endereço da pilha indicado pelo SP: o conteúdo atual do SP é decrementado - 1 e aí copiado o byte superior do registro, o conteúdo do SP é novamente decrementado - 1 e aí copiado o byte inferior do registro. Ou seja, se o conteúdo atual do SP indica (nn) o byte superior será copiado em (nn-1) e o byte inferior em (nn-2).

Para os registros de índice usa-se a instrução "PUSH rr", código 11-1 1101/1110 0101, com IX=0 e IY=1.

"POP rr", código 11.. 0001, copia no par de registros rr os 2 bytes cujo endereço é indicado pelo SP. O conteúdo atual do SP indica o endereço do byte que será copiado no byte inferior do par de registros, em seguida ele é incrementado + 1 e passa a apontar para o byte que será co-

piado no byte superior do par. Ou seja, se o conteúdo atual do SP indica (nn) este será o endereço correspondente ao byte inferior e (nn+1) ao byte inferior.

O programa 11 carrega 2047 no SP; o par AF (acumulador e registro de flags) é copiado no endereço apontado pelo SP, A em 2046 e F em 2045. SP aponta agora para 2045 e com a instrução "POP DE" o conteúdo de 2045 - o mesmo de F - é carregado em E e o conteúdo de 2046 em D. O byte carregado em E é colocado na porta da saída indicada por C. Com isso conseguimos registrar na porta de saída o estado das flags existente no registro F.

PROGRAMA 11 - Leitura e apresentação das flags por manipulação na pilha

ASSEMBLER

```
LABEL  MNEM.  OPERANDO      COMENTÁRIOS
       ORG    0            ; endereço da instrução ini
                           ; cial
       LD     SP,2047      ; carrega 0000 0111 1111 1111
                           ; B no SP
       PUSH   AF           ; copia registros A e F na
                           ; pilha indicada por SP
       POP    DE           ; carrega da pilha em DE os
                           ; conteúdos de AF copiados
       LD     C,127        ; carrega 0111 1111B em C
       OUT    E,(C)        ; coloca na porta de saída in
                           ; dicada por (C) o conteúdo
                           ; de E
       HALT                ; fim do programa
```

PROGRAMA 11

CÓDIGO DE MÁQUINA

ENDEREÇO NA MEMÓRIA	CÓDIGO	(ASSEMBLER)
0000 0000 0000 0000	0011 0001	LD SP,2047
0001	1111 1111	
0010	0000 0111	
0011	1111 0101	- PUSH AF
0100	1101 0001	- POP DE
0101	0000 1110	LD C,127
0110	0111 1111	
0111	1110 1101	OUT E,(C)
1000	0101 1001	
1001	0111 0110	- HALT

Ao sair do reset o Z-80 gravará nos 8 LEDs o conteúdo do registro de flags F.

9 - INTERRUPÇÕES

O Z-80 atenderá a um sinal de interrupção no pino 16 se estiver previamente programado com a instrução "EI", código 1111 1011. Se ao contrário for acrescentada a instrução "DI", código

1111 0011, ele desconhecerá o sinal de interrupção. Para definir qual modo de interrupção (veja Capítulo 2 - seção 6 - interrupção) será processado usa se a instrução "IM n", onde n pode ser 0,1 ou 2, código 1110 1101/010. ,110; os dois bits vagos são preenchidos com 00=IM0, 10=IM1 e 11=IM2. Antes de iniciar o tratamento da interrupção o Z-80 coloca na pilha (indicada por SP) o conteúdo atual do PC (contador de programa) e reseta a flag I, desabilitando outras interrupções. No fim da rotina de interrupção deve ser programada a instrução "RETI" , código 1110 1101/0100 1101, que repõe no PC o endereço existente no topo da pilha e seta a plag I, habilitando interrupções.

No programa 12 a instrução no endereço zero carrega o acumulador com zero e grava esse número na porta de saída, em seguida habilitando interrupções, selecionando o modo 0 de interrupção e parando; no endereço 8 começa nova parte do programa que carrega o acumulador com 8 e o grava na porta de saída. Após o reset o Z-80 executará a la. parte do programa, gravando zero nos 8 LEDs, ficando no endereço 7 parado e habilitado para interrupção modo 0; se a chave S4 for acionada para baixo - $\overline{INT}$ - ele executará a instrução cujo código estiver assentado nas chaves de dados. Usaremos "REST p", código 11...111, que carrega o conteúdo atual do PC na pilha indicada por SP e faz o microprocessador executar a rotina num dos 8 endereços selecionado pelos três bits vagos dentro da seguinte correspondência: 000-0/001-8/010-10/011-18/ 100-20/101-28/110-30/111-38.

PROGRAMA 12 - Seleção de rotina de interrupção

ASSEMBLER

```
LABEL  MNEM.  OPERANDO      COMENTÁRIOS
RSTO   ORG      0       ; endereço da 1a. instrução
       LD       A,0     ; carrega 0000 0000B no acu-
                        ; mulador
       OUT      (127),A ; coloca 0000 0000B na porta
                        ; de saída
       EI               ; habilita interrupção
       IMO              ; seleciona interrupção modo
                        ; 0
       HALT             ; parada até que ocorra reset
                        ; ou interrupção
RST8   LD       A,8     ; carrega 0000 1010B no acu
                        ; mulador
       OUT      (127),A ; coloca 0000 1010B na porta
                        ; de saída
       HALT             ; fim do programa ou parada
                        ; até que ocorra reset ou
                        ; interrupção
```

PROGRAMA 12

CÓDIGO DE MÁQUINA

ENDEREÇO NA MEMÓRIA	CÓDIGO	(ASSEMBLER)
0000 0000 0000 0000	0011 1110	LD A,0
0001	0000 0000	
0010	1101 0011	OUT (127),A
0011	0111 1111	
0100	1111 1011	- EI
0101	1110 1101	IMO
0110	0100 0110	
0111	0111 0110	- HALT
1000	0011 1110	LD A,8
1001	0000 1010	
1010	1101 0011	OUT (127),A
1011	0111 1111	
1100	0111 0110	- HALT

Após sair do reset o Z-80 coloca "0000 0000" nos 8 LEDs. Assente a instrução "RST8" 1100 1111 - nas chaves de dados e acione a chave S4 para baixo ($\overline{INT}$); o Z-80 irá executar a rotina do endereço 8, gravando "0000 1010" nos 8 LEDs e parando no endereço 12. Se agora for posto "RSTO" nas chaves de dados e acionada S4 ele não obedecerá a interrupção, pois ao fazê-lo na primeira desabilitou a flag I e nenhuma instrução voltou a habilitá-la.

Na configugação básica do MICRO-GALENA não é possível executar a interrupção não mascarável pois o pino 17 - $\overline{NMI}$ - do Z-80 é mantido desativado permanentemente (para experiência o resistor pode ser desviado para o pino 16 - $\overline{INT}$ - e a saída da chave S-4 soldada ao pino 17 - $\overline{NMI}$). A operação é semelhante à interrupção comum, porém

ela desconhece a programação de "EI" ou "DI"; a flag I não é posteriormente resetada como na interrupção comum, mas copiada numa segunda flag IFF2. No fim da rotina de interrupção não mascarável a instrução "RETN", código 1110 1101/0100 0101, tira da pilha indicada por SP o endereço da instrução que estava para ser executada no momento da interrupção e o repõe no PC, além de copiar em IFF1 o conteúdo de IFF2, recolocando a flag I no mesmo estado anterior à interrupção.

10 - ENTRADA E SAÍDA DE BLOCOS DE DADOS

"INI", código 1110 1101/1010 0010, recebe um byte da porta indicada pelo registro C e o copia no endereço apontado por HL, incrementando +1 o conteúdo deste último par e decrementando -1 o registro B. "IND", código 1110 1101/1010 1010, executa a mesma operação, exceto que o par HL é decrementado - 1 ao invés de ser incrementado.

"OUTI", código 1110 1101/1010 0011, coloca na porta de saída indicada pelo registro C o conteúdo do endereço apontado por HL, incrementando + 1 este último par e decrementando o registro B. "OUT D", código 1110 1101/1010 1011, executa a mesma operação, exceto que o par HL é decrementado -1 ao invés de ser incrementado.

No programa 13 o par HL é carregado com 256 - portanto o registro L terá como conteúdo 0000 0000B - e o registro B com 4; após uma entrada de dados usando "INI" o conteúdo de 1 é colocado na porta de saída indicada por C e o conteúdo de B copiado endereço indicado por HL - que será 256+ 1.

PROGRAMA 13 - Verificação do incremento de HL na instrução "INI"

ASSEMBLER

```
LABEL  MNEM.   OPERANDO        COMENTÁRIOS

       ORG       0          ; endereço da 1a. instrução
       LD      HL,256       ; carrega H com 0000 0001B e
                            ; L com 0000 0000B
       LD        B,4        ; carrega B com 0000 0100B
       INI                  ; dá entrada em 1 byte
       LD      (HL),B       ; copia conteúdo de B no ende
                            ; reço indicado por (HL)
       LD        C,127      ; carrega 0111 1111B em C
       OUT     (C), L       ; coloca conteúdo de L na
                            ; porta de saída
       HALT                 ; fim do programa
```

PROGRAMA 13

CÓDIGO DE MÁQUINA

ENDEREÇO NA MEMÓRIA	CÓDIGO	(ASSEMBLER)
0000 0000 0000 0000	0010 0001	
0001	0000 0000	LD HL,256
0010	0000 0001	
0011	0000 0110	LD B,4
0100	0000 0100	
0101	1110 1101	INI
0110	1010 0010	
0111	0111 0000	-LD(HL),B
1000	0000 1110	LD C,127
1001	0111 1111	
1010	1110 1101	OUT(C),L
1011	0110 1001	
1100	0111 0110	-HALT

Durante o reset é posto um byte nas chaves de dados. Ao sair do reset o Z-80 escreverá "0000 0001" nos 8 LEDs, que é o conteúdo do registro L correspondendo ao incremento 256+1=257 0000 0001/0000 0001B. Se for solicitada DMA e com a ponta de prova for feita a leitura do conteúdo do endereço 256 (0000 0001 0000 0000B) será encontrado o byte posto na chave de dados. No endereço seguinte (256+1=257) estará o byte 0000 0101; que é o conteúdo de B decrementado pela instrução "INI".

As instruções "INIR" - 1110 1101/1011 0010, "INDR" - 1110 1101/1011 1010, "OTIR" 1110 1101/1011 0011, "OTDR" - 1110 1101/1011 1011, executam as mesmas operações que suas correspondentes "INI", "IND", "OUTI", "OUTD", respectivamente, exceto que continuam a entrar/sair novos bytes, incrementando/decrementando HL e decrementando B, até que o conteúdo de B seja zero.

11 - TRANSFERÊNCIA DE BLOCO DE DADOS

"LDI", código 1110 1101/1010 0000, copia no endereço indicado por (DE) o conteúdo da posição apontada por (HL); após a transferência DE e HL são incrementados + 1 e o par BC decrementado -1. "LDD", código 1110 1101/1010 1000, executa a mesma operação acima, porém DE e HL são decrementados -1 ao invés de serem incrementados.

No programa 14 HL é carregado com o endereço fonte 7 e DE com o endereço destino 21, enquanto BC é carregado com a contagem inicial 15; segue-se a instrução "LDI" que irá copiar no endereço 21 o conteúdo da posição 7. O acumulador copia o conteúdo do endereço 21 - que será a copia do

endereço 7 - e o coloca na porta de saída. O registro C - parte inferior do par BC que continha 15 e foi decrementado para 14 - é copiado no endereço 22.

PROGRAMA 14 - Transferência de dados usando os ponteiros HL e DE

ASSEMBLER

```
LABEL  MNEM.  OPERANDO      COMENTÁRIOS
       ORG      0          ; endereço da 1a. instrução
       LD      HL,7        ; carrega endereço fonte 000
                           ; 0000/0000 0111B
       LD      DE,21       ; carrega endereço destino
                           ; 0000 0000/0001 0101B 21 em
                           ; DE
       LD      BC,15       ; carrega contagem inicial
                           ; 0000 0000/0000 1111B em BC
       LDI                 ; copia byte existente no en
                           ; dereço 7 no endereço 21; in
                           ; crementa HL e DE e decremen
                           ; ta BC
       LD       A,(21)     ; carrega no acumulador con
                           ; teúdo do endereço 21
       OUT    (127),A      ; coloca conteúdo do acumula
                           ; dor na porta de saída
       LD     (22), C      ; carrega conteúdo de C no
                           ; endereço 0000 0000/ 0001
                           ; 0110B
       HALT                ; fim do programa
```

PROGRAMA 14

CÓDIGO DE MÁQUINA

ENDEREÇO NA MEMÓRIA	CÓDIGO	(ASSEMBLER)
0000 0000 0000 0000	0010 0001	
0001	0000 0111	LD HL,7
0010	0000 0000	
0011	0001 0001	
0100	0001 0101	LD DE,21
0101	0000 0000	
0110	0000 0001	LD BC,15
0000 0000 0000 0111	0000 1111	
1000	0000 0000	
1001	1110 1101	LDI
1010	1010 0000	
1011	0011 1010	LD A,(21)
1100	0001 0101	
1101	0000 0000	
1110	1101 0011	OUT (127),A
1111	0111 1111	
0001 0000	1110 1101	
0001 0001	0100 0011	LD (22),C
0001 0010	0001 0110	
0001 0011	0000 0000	
0001 0100	0111 0110	- HALT

Após sair do reset o Z-80 escreverá "0000 1111" nos 8 LEDs, correspondente ao byte gravado no endereço 7 e que foi copiado no endereço 21. Pedindo uma DMA e fazendo com a ponta de prova a leitura do endereço 22 o operador deverá encontrar "0000 1110", referente ao byte inferior do par BC decrementado.

As instruções "LDIR", código 1110 1101/1011 0000, e "LDDR", código 1110 1101/ 1011 1000, executam as mesmas operações que "LDI" e "LDD", respectivamente, porém continuam transferindo bytes até que BC seja zero; cada nova transferência usa os novos endereços incrementados ou decrementados na vez anterior.

12 - TROCA ENTRE CONTEÚDO DE REGISTROS

Os conteúdos dos registros DE e HL podem ser trocados entre si pela instrução "EX DE, HL", código 1110 1011.

"EX (SP), HL" troca o conteúdo de L com o conteúdo do endereço na pilha indicado por (SP), incrementa SP e troca o conteúdo de H com o conteúdo do novo endereço incrementado (SP+1). O código desta instrução é 1110 0011. Este mesmo esquema de troca é usado na instrução "EX (SP), rr" , código 11.1 1101/1110 0011, preenchendo-se o bit vago com 0 para IX e 1 para IY. Agora o conteúdo de (SP) é trocado com o byte inferior de IX ou IY

e (SP+1) com o byte superior.

No Capítulo 2 vimos que o Z-80 dispõe de um conjunto principal de registros - A,F,B, C,D,E,H,L - e outro alternativo - A', F', B', C', D', E', H', L'. Quando um conjunto está sendo usado o outro fica na reserva, ocioso.

"EX AF, AF'", código 0000 1000, coloca na reserva os dois registros A e F que estavam sendo usados e passa a empregar seus equivalentes A' e F' que se encontravam ociosas. "EXX", código 1101 1001, troca de uma só vez os pares restantes: BC-B'C', DE-D'E', HL-H'L'. Não há distinção alguma entre um registro principal e seu equivalente alternativo; quando é executada a instrução "EXX" ou "EX AF, AF'" o principal passa a ser o alternativo e vice-versa.

No programa 15 o par HL é carregado com 0=0000 0000 0000 0000B - portanto o registro L conterá 0000 000B; chamaremos este par de 1º HL. A instrução "EXX" troca o conjunto principal pelo alternativo, colocando na reserva o 1º HL e em atividade o HL alternativo, que chamaremos de 2º HL. O 2º HL é carregado com 15=0000 0000/0000 1111B em conseqüência o registro 2º L estará carregado com 0000 1111B e é colocado na porta de saída. Seleciona-se a interrupção modo 1 (desvio para endereço 56), o SP é carregado para iniciar a pilha na posição 2047, habilita-se interrupções e é feita a parada. No endereço 56 de interrupção modo 1 inicia-se uma rotina que troca o atual conjunto principal pelo alternativo - logo o 2º HL é levado para a reserva e o 1º HL contendo 0000 0000 0000 0000B é posto em atividade. A instrução "OUT (C), L" gravará o conteúdo de L na porta de saída, mas desta vez

será o 1º L contendo 0000 0000B.

PROGRAMA 15 - Comparação dos conteúdos de registros principais e alternados

ASSEMBLER

```
LABEL   MNEM.   OPERANDO        COMENTÁRIOS
        ORG       0         ; endereço da 1a. instrução
                            ; do programa
        LD       HL,0       ; carrega 0000 0000     0000
                            ; 0000B no 1º HL
        EXX                 ; troca conjunto  principal
                            ; por alternativo,  ativando
                            ; 2º HL
        LD       HL,15      ; carrega 0000 0000     0000
                            ; 1111B no 2º HL
        LD        C,127     ; carrega 0111 1111B em C
        OUT      (C), L     ; conteúdo do 2º L = 0000
                            ; 1111B é colocado na  porta
                            ; de saída
        IM1                 ; seleciona interrupção modo
                            ; 1
        LD SP, 2047         ; inicia pilha no  endereço
                            ; 2047
        EI                  ; habilita interrupção
        HALT                ; parada
        ORG       56        ; endereço da 1a.  instrução
                            ; da rotina de interrupção
        EXX                 ; troca conjunto principal
                            ; por alternativo, ativando
                            ; 1º HL
        OUT      (C),L      ; conteúdo do 1º L = 0000
                            ; 0000B é colocado na  porta
                            ; de saída
        HALT                ; fim da rotina de interrup-
                            ; ção
```

PROGRAMA 15

CÓDIGO DE MÁQUINA

ENDEREÇO NA MEMÓRIA	CÓDIGO	(ASSEMBLER)
0000 0000 0000 0000	0010 0001	
0001	0000 0000	LD HL,0
0010	0000 0000	
0011	1101 1001	- EXX
0100	0010 0001	
0101	0000 1111	LD HL,15
0110	0000 0000	
0111	0000 1110	LD C,127
1000	0111 1111	
1001	1110 1101	OUT (C),L
1010	0110 1001	
1011	1110 1101	IM1
1100	0101 0110	
1101	0011 0001	
1110	1111 1111	LD SP,2047
1111	0000 0111	
0001 0000	1111 1011	- EI
0001 0001	0111 0110	- HALT
0000 0000 0111 1000	1101 1001	- EXX
0000 0000 0111 1001	1110 1101	OUT (C),L
0000 0000 0111 1010	0110 1001	
0000 0000 0111 1011	0111 0110	- HALT

Após o reset o Z-80 gravará "0000 1111" nos 8 LEDs, correspondendo ao conteúdo do 2º L. Acionando a chave S-4 para sinalizar pedido de interrupção o Z-80 a reconhecerá, executando a rotina gravada no endereço 56 que coloca nos 8 LEDs o conteúdo "0000 0000" do 1º L.

13 - SET E RESET DE BIT NO REGISTRO

"SET b,r", código 1100 1011/ 11bb b..., carrega 1 (seta) na posição b do registro r; b é um binário de 000B a 111B, selecionando uma das oito posições do conteúdo do registro, começando do LSB à direita (b=000B) até o MSB à esquerda (b=111B); os três espaços no final são preenchidos com o código do registro (B=000, C=001, D=010, E=011, H=100, L=101, (HL)=110, A=111). "RES b, r", código 1100 1011/10bb b..., executa a mesma operação mas zerando (resetando) a posição selecionada. "SET b, (rr+d)", código 11.1 1101/1100 1011/dddd dddd/11bbb110, e "RES b, (rr+d)", código 11.1 1101/ 1100 1011/dddd dddd/10bbb110, setam ou resetam, respectivamente, o bit b do conteúdo do endereço obtido pela soma de IX ou IY com o deslocamento d, sendo este último um número complemento dois; preenchendo se o bit vago no início com 1 seleciona-se IY, e com zero IX.

Um detalhe importante nestas instruções é que apenas o bit setado ou resetado tem valor definido, os outros bits do registro assumem valores aleatórios.

No programa 16 um byte da porta de entrada é copiado no acumulador, seu bit 7 (O MSB, extrema esquerda) é setado e o bit 3 (a 4a. posição a partir da direita) resetado; o byte assim alterado é gravado na porta de saída.

PROGRAMA 16 - Set e reset de bit do acumulador

ASSEMBLER

```
LABEL  MNEM.  OPERANDO        COMENTÁRIOS
       ORG      0       ; endereço da 1a. instrução
       IN     A,(255)   ; byte entra pela porta e é
                        ; carregado no acumulador
       SET    7,A       ; seta MSB do acumulador
       RES    3,A       ; reseta bit 3 do acumulador
       OUT    (127),A   ; coloca acumulador na porta
                        ; de saída
       HALT             ; fim do programa
```

PROGRAMA 16

CÓDIGO DE MÁQUINA

ENDEREÇO NA MEMÓRIA	CÓDIGO	(ASSEMBLER)
0000 0000 0000 0000	1110 1101	
0001	0111 1000	IN A, (255)
0010	1111 1111	
0011	1100 1011	
0100	1111 1111	SET 7,A
0101	1100 1011	
0110	1001 1111	RES 3,A
0111	1101 0011	
1000	0111 1111	OUT (127),A
1001	0111 0110	- HALT

Coloque um byte nas chaves de dados. Ao sair do reset o Z-80 gravará 1XXX 0XXX, que é este byte devidamente alterado. Coloque outro byte e faça novo reset, observando que o resultado sem-

pre será 1XXX 0XXX.

14 - OPERAÇÕES COM BIT DE FLAGS

Vimos no Capítulo 2 que o registro status (S,Z,-,H,-,P/V,N,C) tem suas flags alteradas conforme a instrução que é executada. Entretanto, três instruções permitem operar diretamente as flags Z e C.

"SCF", código 0011 0111, seta (carrega 1) o flag C. "CCF", código 0011 1111, complementa o flag C, isto é, carrega 1 se ele estiver em zero e vice-versa.

"BIT b,r", código 1100 1011/01bbb... carrega no flag Z o complemento do bit b do registro r, onde b é um número binário indicando a posição do bit (000B = LSB; 111=MSB). Para os registros de índice esta instrução é "BIT b, (IX+d)", código 11,1 1101/1100 1011/dddd dddd/01bbb 110, complementando na flag Z o bit b do endereço obtido pela soma de IX ou IY com d, sendo este último um número complemento 2; preenchendo o bit vago no início com zero seleciona-se IX e com 1 IY.

No programa 17 um byte entra pela porta indicada por (C) e é copiado no acumulador; seu bit 5 (o 6º da esquerda para a direita) é testado, gravando-se seu complemento na flag Z. A flag C é setada. O SP indica 2047 como início da pilha, para a qual é enviado o par AF; A ficará em 2047 e F em 2046. A instrução "POP HL" copia o conteúdo superior da pilha - 2047, o acumulador - em H e o conteúdo inferior - 2046, o registro F de flags - em L. Temos agora o conteúdo do registro de

flags transferido para o registro L, que é gravado na porta de saída 127 indicada por C.

PROGRAMA 17 - Operação e leitura de flags C e Z

ASSEMBLER

```
LABEL  MNEM.   OPERANDO      COMENTÁRIOS

       ORG       0       ; endereço da 1a. instrução
       IN      A,(C)     ; copia porta de entrada no
                         ; acumulador
       BIT       5,A     ; grava na flag Z complemento
                         ; do bit 5 do acumulador
       SCF               ; carrega 1-seta-na flag C
       LD      SP,2047   ; inicia a pilha em 2047 = 111
                         ; 1111 1111B
       PUSH      AF      ; coloca A em 2047 e F em 2046
       POP       HL      ; copia conteúdo de 2047 em H
                         ; e conteúdo de 2046 em L
       LD        C,127   ; carrega 0111 1111B em C
       OUT      (C),L    ; coloca conteúdo de L na por
                         ; ta de saída
       HALT              ; fim do programa
```

PROGRAMA 17

CÓDIGO DE MÁQUINA

ENDEREÇO NA MEMÓRIA	CÓDIGO	(ASSEMBLER)
0000 0000 0000 0000	1110 1101	IN A,(C)
0001	0111 1000	
0010	1100 1011	BIT 5,A
0011	0110 1111	
0100	0011 0111	- SCF

ENDEREÇO NA MEMÓRIA	CÓDIGO	(ASSEMBLER)
0101	0011 0001	
0110	1111 1111	LD SP,2047
0111	0000 0111	
1000	1111 0101	- PUSH AF
1001	1110 0001	- POP HL
1010	0000 1110	LD C, 127
1011	0111 1111	
1100	1110 1101	OUT (C),L
1101	0110 1001	
1110	0111 0110	- HALT

Enquanto o Z-80 estiver no reset coloque um byte na chave de dados. Ao sair do reset o Z-80 gravará 1 no LED de D0, referente ao registro da flag C, e no LED de D6 o complemento do bit assentado na 6a. chave (correspondendo a D5), que foi gravado na flag Z. Por exemplo, se o byte for 0010 0000 aparecerá X0XXX XXX1, onde zero refere-se à flag Z e 1 à flag C.

15 - INSTRUÇÕES DE DESVIO (JUMP)

O Z-80 lê e executa as instruções gravadas na memória, uma a uma em ordem crescente, começando daquela gravada no endereço zero. As instruções de desvio (jump) alteram essa seqüência, carregando um novo endereço no PC - contador de programa.

"JP nn", código 1100 0011/.... ..../ ...., desvia o programa para o endereço preenchido nos dois últimos bytes (primeiro inferior, depois o superior), "JP (HL)", código 1110 1001, desvia o programa para o endereço igual ao conteú

do do par HL. "JP (rr)", código 11.1 1101/1110 1001 desvia para o endereço igual ao conteúdo de um dos registros de índice, preenchendo o bit vago com zero para IX e 1 para IY.

"JR d", código 0001 1000/dddd dddd, desvia para o deslocamento d. Este é um número complemento 2, indo de -127=1000 0001B até +127 = 0111 1111; valores positivos indicam deslocamento crescente (para frente) contados a partir do fim da instrução, valores negativos indicam deslocamento para trás (o PC é somado ou diminuido de d). Assim, se a instrução "JR + 5" for gravada nos endereços 21 e 22 o desvio será para a posição 27; a instrução "JR - 5" gravada na mesma posição que a anterior desviará para 18.

"JP cc, nn", código 11cc c010/ ..../.... ...., desvia para o endereço nn apenas se a condição ccc for verdadeira, caso contrário será executada a instrução seguinte na seqüência comum. ccc são três bits com a correspondência abaixo (na linguagem assembler ccc é substituído pelo mnemônico da condição):

CCC	MNEM.	CONDIÇÃO
000	NZ	flag Z=0
001	Z	flag Z=1
010	NC	flag C=0
011	C	flag C=1
100	PO	flag P/V=0
101	PE	flag P/V=1
110	P	flag S=0
111	M	flag S=1

"JR cc,d", código 001c c000/ dddd dddd, desvia para o deslocamento d (com as mesmas especificações vistas para "JRd") se for verdadeira a condição cc dada na tabela abaixo:

CC	MNEM	CONDIÇÃO
00	NZ	flag Z=0
01	Z	flag Z=1
10	NC	flag C=0
11	C	flag C=1

"DJNZ d", código 0001 0000/dddd dddd decrementa o registro B e desvia para o deslocamento d (com as mesmas especificações vistas para "JRd" e "JRcc,d"), repetindo a operação até que B=0, quando então termina os desvios e a instrução seguinte na seqüência comum é processada.

No programa 18 um byte entra pela porta apontada por C e é carregado no registro H, sendo seu bit 7 (0 MSB) testado: se for 1 a flag Z é resetada (zerada) e o programa é desviado para a rotina NEG, que carrega 1000 000B no registro D e o grava na porta de saída; se o MSB de H for zero o programa desconhece este primeiro desvio para NEG e executa a próxima instrução, que é um desvio relativo para a rotina POS, onde o registro D é carregado com 0111 1111 e gravado na porta de saída. Observe que na linguagem assembler os endereços para desvio são substituídos pelo "LABEL" correspondente.

PROGRAMA 18 - Desvio (jump) da seqüência do programa

ASSEMBLER

```
LABEL   MMEM.   OPERANDO        COMENTÁRIOS
        ORG       0       ; endereço da 1a. instrução
        LD      C,127     ; carrega 0111 1111B em C
        IN      H,(C)     ; copia em H byte da porta
                          ; apontada por C
        BIT     7,H       ; testa MSB de H,     setando
                          ; ou resetando flag Z
        JPNZ    NEG       ; desvia para NEG se flag  Z=
                          ; 0; caso contrário executa a
                          ; próxima instrução
        JR      POS       ; desvia para POS, localizada
                          ; entre - 126 e +127 desta ins
                          ; trução; o desvio é d=+6
        HALT              ; parada
NEG     ORG     1024      ; endereço da 1a. instrução
                          ; da rotina NEG
        LD      D,128     ; carrega 1000 0000B em D
        OUT     (C),D     ; grava conteúdo de D na  por
                          ; ta de saída
        HALT              ; parada
POS     ORG       16      ; endereço da 1a. instrução
                          ; da rotina POS
        LD      D,127     ; carrega 0111 1111B em D
        OUT     (C), D    ; grava conteúdo de D na  por
                          ; ta de saída
        HALT              ; parada
```

PROGRAMA 18

CÓDIGO DE MÁQUINA

ENDEREÇO NA MEMÓRIA	CÓDIGO	(ASSEMBLER)
0000 0000 0000 0000	0000 1110	LD C, 127
0001	0111 1111	
0010	1110 1101	IN H, (C)
0011	0110 0000	
0100	1100 1011	BIT 7,H
0101	0111 1100	
0110	1100 0010	
0111	0000 0000	JPNZ NEG
1000	0000 0100	
1001	0001 1000	JR POS
1010	0000 0110	
1011	0111 0110	- HALT
0000 0000 0001 0000	0001 0110	LD D,127
1 0001	0111 1111	
1 0010	1110 1101	OUT (C),D
1 0011	0101 0001	
1 0100	0111 0110	- HALT
0000 0100 0000 0000	0001 0110	LD D,128
100 0000 0001	1000 0000	
100 0000 0010	1110 1101	OUT (C),D
100 0000 0011	0101 0001	
100 0000 0100	0111 0110	- HALT

Durante o reset do Z-80 coloque um byte nas chaves de dados. Se seu MSB for 1 será gravado "1000 0000" nos 8 LEDs, se for zero a gravação será "0111 1111".

16 - INSTRUÇÕES DE COMPARAÇÃO

Nas instruções de comparação dois operandos P e Q são confrontados, subtraindo-se o 2º do 1º (P-Q); na realidade a subtração não é realizada e eles permanecem inalterados, mas o registro de flags é alterado como se isso ocorresse:

S: setado (1) se o 2º operando é maior que o 1º (P-Q = negativo)
Z: setado (1) se os dois operando são iguais (P-Q = zero)
H: setado (1) se houver half-carry (P-Q produz "vem um" do bit 4)
C: setado (1) se não houver carry (P-Q produz "vem um" do bit 7)
P/V: setado (1) se houver excesso (P-Q produz resultado que excede o registro).

"CPr", código 1011 1..., compara o acumulador com o conteúdo de um registro selecionado pelos três bits no final. "CPn", código 1111 1110/nnnn nnnn, compara o acumulador com o byte n. "CP (rr+d)", código 11.1 1101/1011 1110/dddd dddd, compara o acumulador com o conteúdo do endereço indicado pelo registro de índice somado ao deslocamento d; o bit vago no início seleciona IX quando for zero ou IY quando for 1; d é um número complemento 2.

"CPI", código 1110 1101/1010 0001, compara o acumulador com o conteúdo de endereço apontado por HL (ou seja, A-(HL)), incrementa +1 HL e decrementa - 1 BC. "CPD", código 1110 1101/ 1010 1001, executa a mesma operação acima mas o par HL é decrementado - 1 ao invés de ser incrementado.

"CPIR", código 1110 1101/1011 0001, compara o acumulador com o conteúdo do endereço apontado por HL, incrementa HL + 1, decrementa BC - 1, repetindo essas operações até que A = (HL) ou BC = 0. "CPDR", código 1110 1101/1011 1001, é semelhante à instrução CPIR, mas o par HL é decrementado - 1 ao invés de ser incrementado.

No programa 19 o registro C é carregado com 0111 1111B, o acumulador com 1111 1110B e L com um byte da porta de entrada. O acumulador é comparado com L, se houver igualdade é executada a rotina IG que coloca o conteúdo 1111 1110B do acumulador na porta de saída, caso contrário o conteúdo 0111 1111B de C é gravado na porta de saída.

PROGRAMA 19 - Comparação do conteúdo do acumulador com registro

ASSEMBLER

```
LABEL   MNEM.   OPERANDO        COMENTÁRIOS
        ORG       0       ; endereço da 1a. instrução
        LD      C,127     ; carrega 0111 1111B em C
        IN      L,(C)     ; copia um byte da porta de
                          ; entrada em L
        LD      A,254     ; carrega 1111 1110B no acu-
                          ; mulador
        CP      L         ; compara A com L, setando
                          ; Z se forem iguais ou rese
                          ; tando se forem diferentes
        JPNZ    DIF       ; desvia para DIF se Z=0; A≠L
IG      OUT     (C),A     ; coloca conteúdo de A na por
                          ; ta de saída
        HALT              ; parada
DIF     OUT     (C),C     ; coloca conteúdo de C na
                          ; porta de saída
        HALT              ; fim do programa
```

PROGRAMA 19

CÓDIGO DE MÁQUINA

ENDEREÇO NA MEMÓRIA	CÓDIGO	(ASSEMBLER)
0000 0000 0000 0000	0000 1110	LD C,127
0001	0111 1111	
0010	1110 1101	IN L,(C)
0011	0110 1000	
0100	0011 1110	LD A,254
0101	1111 1110	
0110	1011 1101	- CP L
0111	1100 0010	
1000	0000 1101	JPNZ DIF
1001	0000 0000	
1010	1110 1101	OUT (C),A
1011	0111 1001	
1100	0111 0110	- HALT
1101	1110 1101	OUT (C),C
1110	0100 1001	
1111	0111 0110	- HALT

Durante o reset coloque um byte na chave de dados. Se for 1111 1110B (igual ao conteúdo do acumulador) após o reset ele mesmo será gravado nos 8 LEDs, caso contrário a gravação será 0111 1111B (conteúdo de C). Repita para vários bytes.

17 - INSTRUÇÕES LÓGICAS

Três funções lógicas - AND, XOR, OR- podem ser executadas entre um operando e o acumulador, ficando o resultado neste último. O registro de flags também é afetado (sobre funções lógicas veja o Capítulo 1).

"ANDr", código 1010 0..., usa como operando o conteúdo do registro selecionado pelos três bits vagos (B=000, C=001, D=010, E=011, H=100, L=101, (HL)=110, A=111). "AND n", código 1110 0110/nnnn nnnn, opera sobre o número imediato especificado no 2º byte da instrução. "AND (rr+d)" código 11.1 1101/1010 0110/dddd dddd, opera sobre o conteúdo do endereço obtido pela soma de um dos registros de índice com o deslocamento d, que é um número complemento 2; zero no bit vago seleciona IX e 1 seleciona IY.

"ORr", código 1011 0..., usa como operando o conteúdo do registro selecionado pelos três bits vagos. "OR n", código 1111 0110/nnnn nnnn, opera sobre o número imediato especificado no 2º byte da instrução. "OR (rr+d)", código 11.1 1101/1011 0110/dddd dddd, opera sobre o conteúdo do endereço obtido pela soma de um dos registros de índice com o deslocamento d, que é um número complemento 2; zero no bit vago seleciona IX e 1 IY.

"XOR r", código 1010 1..., usa como operando o conteúdo do registro selecionado pelos três bits vagos. "XOR n", código 1110 1110/ nnnn, nnnn, opera sobre o número imediato especificado no 2º byte da instrução. "XOR (rr+d)", código 11.1 1101/1010 1110/dddd dddd, opera sobre o conteúdo do endereço obtido pela soma de um dos registros de índice com o deslocamento d, que é um número complemento 2; zero no bit vago seleciona IX e 1IY.

No programa 20 o acumulador é carregado com 0000 1111B. Um byte recebido da porta de entrada é copiado em B. Executa-se a operação A AND B e o resultado, armazenado em A é colocado na porta de saída.

PROGRAMA 20 - Operação lógica AND com o acumulador

ASSEMBLER

```
LABEL   MNEM.   OPERANDO        COMENTÁRIOS
        ORG       0        ; endereço da 1a. instrução
        IN      B,(C)      ; byte da porta de entrada
                           ; é copiado em B
        LD      A, 15      ; carrega 0000 1111B no
                           ; acumulador
        AND       B        ; A and B, resultado fica em
                           ; A
        OUT     (127),A    ; conteúdo do acumulador é
                           ; colocado na porta de saída
        HALT               ; fim do programa
```

PROGRAMA 20

CÓDIGO DE MÁQUINA

ENDEREÇO NA MEMÓRIA	CÓDIGO	(ASSEMBLER)
0000 0000 0000 0000	1110 1101	IN B, (C)
0001	0100 0000	
0010	0011 1110	LD A, 15
0011	0000 1111	
0100	1010 0000	- AND B
0101	1101 0011	OUT (127),A
0110	0111 1111	
0111	0111 0110	- HALT

Durante o reset coloque um byte XXXX XXXX nas chaves de dados. Após o reset aparecerá nos 8 LEDs o resultado da operação XXXX XXXX AND 0000 1111. Repita para vários bytes.

18 - COMPLEMENTAÇÃO DO ACUMULADOR

"CPL", código 0010 1111, inverte os bits do acumulador, transformando 1 em zero e vice-versa (este é o complemento 1); o resultado é armazenado no próprio acumulador.

"NEG", código 1110 1101/0100 0100, subtrái o conteúdo do acumulador de 256 = 1 0000 0000B (este é o complemento 2); o resultado é armazenado no próprio acumulador.

No programa 21 um byte da porta de entrada é carregado no acumulador. A pilha do SP é iniciada em 2047, o modo de interrupção zero selecionado e habita-se interrupções. No endereço 16 é gravada a rotina CPL 1 que complementa o acumulador e coloca o resultado na porta de saída. No endereço 24 é gravada a rotina CPL2 que transforma em complemento 2 o conteúdo do acumulador, colocando o resultado na porta de saída.

PROGRAMA 21 - Complemento 1 e complemento 2 do acumulador

ASSEMBLER

```
LABEL   MNEM.   OPERANDO      COMENTÁRIOS
        ORG       0       ; endereço da 1a. instrução
        IN      A,(C)     ; copia porta de entrada no
                          ; acumulador
        LD      SP,2047   ; aponta pilha em 0000 0111/
                          ; 1111 1111B
        IMO               ; seleciona interrupção mo
                          ; do zero
        EI                ; habilita interrupção
        HALT              ; parada até receber inter
                          ; rupção
CPL1    ORG       16      ; endereço da 1a. instrução
                          ; da rotina CPL1
        CPL               ; complemento 1 do acumula-
                          ; dor
        OUT     (127),A   ; coloca conteúdo complemen
                          ; tado 1 do acumulador na
                          ; porta de saída
        HALT              ; fim da rotina CPL1
CPL2    ORG       24      ; início da rotina CPL2
        NEG               ; complemento 2 do acumula-
                          ; dor
        OUT     (127),A   ; coloca conteúdo complemen
                          ; tado 2 do acumulador na
                          ; porta de saída
        HALT              ; fim da rotina CPL2
```

PROGRAMA 21

CÓDIGO DE MÁQUINA

ENDEREÇO NA MEMÓRIA	CÓDIGO	(ASSEMBLER)
0000 0000 0000 0000	1110 1101	IN A, (C)
0001	0111 1000	
0010	0011 0001	
0011	1111 1111	LD SP, 2047
0100	0000 0111	
0101	1110 1101	IMO
0110	0100 0110	
0111	1111 1011	- EI
1000	0111 0110	- HALT
0000 0000 0001 0000	0010 1111	- CPL
1 0001	1101 0011	OUT (127), A
1 0010	0111 1111	
1 0011	0111 0110	- HALT
0000 0000 0001 1000	1110 1101	NEG
1 1001	0100 0100	
1 1010	1101 0011	OUT (127),A
1 1011	0111 1111	
1 1100	0111 0110	- HALT

Durante o reset coloque um byte nas chaves de dados. Após o reset assente nas chaves de dados o byte 1101 0111, correspondente ao código da instrução "RST 16"; acione a chave $\overline{INT}$, solicitando interrupção - o Z-80 executará a instrução RST 16, processando a rotina CPL1 e gravando nos 8 LEDs o complemento 1 do byte. Coloque 1101 1111 nas chaves de dados, correspondente ao código da instrução "RST 24"; após acionar a chaves $\overline{INT}$ o Z-80 irá executá-la, processando a rotina CPL2 e gravando nos 8 LEDs o complemento 2 do byte. Repita para vários bytes.

19 - INCREMENTAÇÃO E DECREMENTAÇÃO DE REGISTRO

"INC r", código 00...100, incrementa + 1 o registro selecionado pelos três bits; se for um par de registros a instrução será "INC rr", código 00.. 0011 (BC=00, DE=01, HL=10, SP=11). "DEC r", código 00.. .101, e "DEC rr", código 00.. 1011, são as respectivas instruções que decrementam -1.

"INC (rr+d)", código 11.1 1101/ 0011 0100/dddd dddd, incrementa +1 o conteúdo do endereço obtido pela soma de um dos registros de índice com o deslocamento d; zero no bit inicial vago seleciona IX, 1IY; d é um número complemento 2. "DEC (rr+d)", código 11.1 1101/0011 0101/dddd dddd, é semelhante à instrução anterior mas decrementa -1 ao invés de incrementar.

"INC rr", código 11.1 1101/0010 0011 incrementa +1 o conteúdo de um dos registros de índice, preenchendo-se o bit vago com zero para IX e 1 para IY. "DEC rr", código 11.1 1101/0010 1011, é a instrução equivalente à anterior que decrementa -1 ao invés de incrementar.

No programa 22 B é carregado com 10 e um byte da porta de entrada é carregado no acumulador. A seqüência de instruções NUM: "INCA" "DJNZ NUM" incrementa +1 o byte copiado no acumulador, decrementa -1 o registro B e volta para a instrução NUM, repetindo até que B=0, quando então será executada a instrução seguinte que grava na porta de saída o conteúdo do acumulador (esta seqüência de repetição é chamada "looping" - laço).

PROGRAMA 22 - Looping com incrementação do acumulador

ASSEMBLER

```
LABEL   MNEM.   OPERANDO        COMENTÁRIOS
        ORG       0       ; endereço da 1a. instrução
        LD        B,10    ; carrega 0000 1010B no re
                          ; gistro B
        IN        A,(255) ; carrega no acumulador byte
                          ; da porta de entrada
NUM     INC       A       ; incrementa +1 acumulador
        DJNZ      NUM     ; decrementa -1 B e volta pa
                          ; ra instrução NUM;
                          ; quando B=0 passa para a
                          ; próxima instrução
        OUT       (127),A ; grava acumulador incremen
                          ; tado 10 vezes na porta de
                          ; saída
        HALT              ; fim do programa
```

PROGRAMA 22

CÓDIGO DE MÁQUINA

ENDEREÇO NA MEMÓRIA	CÓDIGO	(ASSEMBLER)
0000 0000 0000 0000	0000 0110	LD B, 10
0001	0000 1010	
0010	1101 1011	IN A, (255)
0011	1111 1111	
0100	0011 1100	-INC A
0101	0001 0000	DJNZ NUM (d=-3)
0110	1111 1101	
0111	1101 0011	OUT (127),A
1000	0111 1111	
1001	0111 0110	-HALT

Durante o reset coloque um byte nas chaves de dados. Após será gravado nos 8 LEDs o byte + 10. Repita para vários bytes. Um caso interessante ocorrerá quando o byte for igual ou maior que 246, ultrapassando a capacidade do acumulador (255= 1111 1111B); uma rotina de leitura do registro das flags indicará que P/V foi setada.

20 - OPERAÇÕES ARITMÉTICAS

Nas operações aritméticas de 8 bits um operando é somado ou subtraído do acumulador e o resultado fica armazenado neste último. "ADD A,r" código 1000 0..., soma o conteúdo do acumulador com o registro r selecionado pelos três bits vagos. "ADD A,n",código 1100 0110/.... ...., soma o acumulador com o byte imediato n. "ADD A,(rr+d)", código 11.1 1101/1000 0110/dddd dddd, soma o acumulador com o conteúdo do endereço obtido pela adição de um dos registros de índice com o deslocamento d; no bit vago no início zero seleciona IX e 1 IY; d é um número complemento 2.

"ADC A,r", código 1000 1..., soma o acumulador com o registro r mais o bit existente no flag C. "ADC A,n", código 1100 1110/.... ...., soma o acumulador com o byte imediato n mais o bit da flag C. "ADC A, (rr+d)", código 11.1 1101/ 1000 1110/dddd dddd, soma o acumulador mais o bit da flag C com o conteúdo do endereço obtido pela adição de um dos registros de índice com o deslocamento d; no bit vago no início zero seleciona IX e IY; d é um número complemento 2.

As operações de soma acima tem as seguintes correspondentes para subtração, o resultado também permanecendo no acumulador: "SUB r", có-

digo 1001 0..., acumulador menos registro; "SUB n", código 1101 0110/.... ...., acumulador menos byte imediato; "SUB (rr+d), código 11.1 1101/1001 0110/ dddd dddd, acumulador menos (IX+d) ou (IY+d); "SBC A,r", código 1001 1..., acumulador menos flag C menos registro; "SBC A;n", código 1101 1110/ ...., acumulador menos flag C menos byte imediato; "SBC A, (rr+d)", código 11.1 1101/1001 1110/ dddd dddd, acumulador menos flag C menos (IX+d) ou (IY +d).

"ADD HL,rr", código 00.. 1001, soma um dos quatro pares de registros selecionados pelos dois bits vagos (BC=00, DE=01, HL=10, SP=11) com o conteúdo de HL; o resultado permanece neste último. "ADC HL,rr", código 1110 1101/01.. 1010, é semelhante à instrução anterior mas somando também a flag C. "SBC HL,rr" código 1110 1101/01.. 0010, subtrái de HL tanto o conteúdo do par de registros rr quanto a flag C.

"ADD IX,rr", código 1101 1101/ 00.. 1001, soma IX com o par de registros selecionado pelos dois bits vagos (BC=00, DE=01, IX=10, SP=11). "ADD IY,rr", código 1111 1101/00.. 1001, soma IY com o par de registros selecionado pelos dois bits vagos (BC=00, DE=01, IY=10, SP=11). Note que cada registro de índice pode ser somado consigo mesmo mas não com seu similar.

No programa 23 um byte da porta de entrada é copiado em B. Seleciona-se o modo 1 de interrupção, a pilha do SP é iniciada em 2047, habilita-se interrupções e o processador fica em HALT até ser interrompido ou resetado. No endereço 56 de interrupção modo 1 uma rotina copia no acumulador um byte da porta de entrada; RETI termina esta rotina, retornando o processamento à instrução se-

guinte àquela em que ocorreu a interrupção. O acumulador é somado com B e o resultado colocado na porta de saída.

PROGRAMA 23 - Soma de dois bytes

ASSEMBLER

```
LABEL   MNEM.   OPERANDO        COMENTÁRIOS
        ORG     0               ; endereço da 1a. instrução
BBYT    IN      B,(C)           ; carrega em B byte da porta
                                ; de entrada
        IM1                     ; seleciona interrupção modo
                                ; 1
        LD      SP,2047         ; inicia pilha em 0000 0111
                                ; 1111 1111B
        EI                      ; habilita interrupção
        HALT                    ; parada até receber inter
                                ; rupção ou reset
SOMA    ADD     A,B             ; soma acumulador com B, o
                                ; resultado fica no acumula-
                                ; dor
        OUT     (127),A         ; coloca conteúdo do acumu
                                ; lador na porta de saída
        HALT                    ; fim do programa
ABYT    ORG     56              ; endereço da rotina de in-
                                ; terrupção modo 1
        IN      A,(C)           ; carrega no acumulador byte
                                ; da porta de entrada
        RETI                    ; termina rotina e volta pa
                                ; ra instrução seguinte à
                                ; quela em que ocorreu a
                                ; interrupção (SOMA)
```

PROGRAMA 23

CÓDIGO DE MÁQUINA

ENDEREÇO NA MEMÓRIA	CÓDIGO	(ASSEMBLER)
0000 0000 0000 0000	1110 1101	IN B, (C)
0001	0100 0000	
0010	1110 1101	IM1
0011	0101 0110	
0100	0011 0001	
0101	1111 1111	LD SP, 2047
0110	0000 0111	
0111	1111 1011	- EI
1000	0111 0110	- HALT
1001	1000 0000	- ADD A,B
1010	1101 0011	OUT (127),A
1011	0111 1111	
1100	0111 0110	- HALT
0000 0000 0011 1000	1110 1101	IN A, (C)
0011 1001	0111 1000	
0011 1010	1110 1101	RETI
0011 1011	0100 1101	

Durante o reset coloque um byte nas chaves de dados; após o reset o Z-80 irá carregá lo no registro B e parar, aguardando interrupção. Coloque o 2º byte nas chaves de dados e acione a interrupção; ele será carregado no acumulador. Saindo da rotina de interrupção o Z-80 somará o acumulador com B, isto é, somará os dois bytes colocados nas chaves de dados e gravará a resposta nos 8 LEDs. Volte ao reset e repita para outro par de bytes. Verifique o que ocorre quando a soma ultrapassa a capacidade do acumulador.

21 - DAA - ARITMÉTICA DE BINÁRIOS CONVERTIDOS PARA BCD

As instruções aritméticas que vimos são válidas para números binários e se aplicadas aos números BCD (decimal codificado em binário) produziriam erros. Como um grande número de interfaces para entrada e saída de dados trabalha com BCD o microprocessador Z-80 executa a instrução "DAA", código 0010 0111B, que automaticamente corrige o resultado binário existente no acumulador, convertendo-o em BCD.

Lembremos do Capítulo 1 que, no BCD, cada dígito decimal é codificado em 4 bits (0=0000, 1=0001, 2=0010, 3=0011, 4=0100, 5=0101, 6=0110, 7=0111, 8=1000, 9=1001). Um BCD tem tantos grupos de 4 bits quanto for o número de dígitos do decimal (23 = 0010 0011, dois dígitos decimais e dois grupos de 4 bits BCD; 523 = 0101 0010 0011, três dígitos decimais e três grupos de 4 bits BCD).

O valor máximo que os 4 bits BCD podem assumir é 1001 = 9, o maior dígito decimal; passando disso não há correspondência com qualquer dígito decimal: 1010 = (10), 1011 = (11), 1100 = (12), 1101 = (13), 1110 = (14), 1111 = (15). Portanto, os números 1010 a 1111 não são válidos em BCD e devem ser corrigidos.

Numa operação aritmética com BCD sendo tratado como binário os dígitos inválidos acima podem ocorrer. Exemplo:

```
+  BCD 24:  0010  0100
+  BCD 38:  0011  1000   +
           ----------
           0101  1100
```

Note no exemplo que o 2º dígito, 1100 = (12), não é válido.

Além disso, toda operação aritmética com BCD que produz um carry ou borrow ("vai um" na soma, "vem um" na subtração) do 4º para o 5º bit apresenta resultado errado. Exemplo:

```
BCD 28:   0010  1000
BCD 39:   0011  1001   +
   67     0110  0001   =  61
CORRETO                INCORRETO
```

A solução do problema, executada pela instrução DAA, é somar 6 = 0110 sempre que o número for um BCD inválido ou que houver carry ou borrow do 4º para o 5º bit.

Somando 6 = 0110 ao BCD inválido resulta em dois dígitos BCD válido:

```
1010 = (10): 1010 + 0110 = 0001 0000
1011 = (11): 1011 + 0110 = 0001 0001
1100 = (12): 1100 + 0110 = 0001 0010
1101 = (13): 1101 + 0110 = 0001 0011
1110 = (14): 1110 + 0110 = 0001 0100
1111 = (15): 1111 + 0110 = 0001 0101
```

O resultado da operação aritmética com carry ou borrow também assume seu valor correto com a soma de 6 = 0110.

```
BCD 28: 0010  1000
BCD 39: 0011  1001   +
   67   0110  0001   (OCORRE CARRY DO 4º PARA o 5º
               0110   BIT)
        0110  0111 = 67
```

As explicações acima são fundamentos teóricos. Para o programador que estiver trabalhando com BCD basta programar "DAA" após cada operação aritmética e o resultado BCD corrigido será obtido.

No programa 24 um byte contendo dois dígitos BCD da porta de entrada é copiado em B. Seleciona-se o modo 1 de interrupção, a pilha do SP é iniciada em 2047, habilita-se interruções e o processador fica em HALT até ser interrompido ou resetado. No endereço 56 de interrupção modo 1 uma rotina copia no acumulador outro byte contendo dois dígitos BCD da porta de entrada; RETI termina esta, retornando o processamento à instrução seguinte àquela em que ocorreu a interrupção. O acumulador é somado com B, o resultado binário é corrigido para BCD pela instrução DAA e colocado na porta de saída.

PROGRAMA 24 - Soma de dois números BCD

ASSEMBLER

```
LABEL   MNEM.   OPERANDO      COMENTÁRIOS
        ORG     0             ; endereço da 1a. instrução
BCD1    IN      B,(C)         ; carrega em B, da porta de
                              ; entrada, 1º byte contendo
                              ; dois dígitos BCD
        IM1                   ; seleciona interrupção mo-
                              ; do 1
        LD      SP, 2047      ; inicia pilha em 0000 0111
                              ; 1111 1111B
        EI                    ; habilita interrupção
        HALT                  ; parada até receber inter
                              ; rupção ou reset
SOMA    ADD     A,B           ; soma acumulador com B, o
                              ; resultado fica no acumula-
                              ; dor
```

```
LABEL   MNEM.   OPERANDO        COMENTÁRIOS
        DAA                     ; converte resultado binário
                                  em BCD
        OUT     (127),A         ; coloca conteúdo do acumu
                                ; lador na porta de saída
        HALT                    ; fim do programa
BCD2    ORG     56              ; endereço da rotina BCD2
        IN      A,(C)           ; carrega no acumulador, da
                                ; porta de entrada, 2º byte
                                ; contendo dois dígitos BCD
        RETI                    ; termina rotina e volta
                                ; para instrução seguinte à
                                ; quela em que ocorreu a in
                                ; terrupção (SOMA)
```

PROGRAMA 24

CÓDIGO DE MÁQUINA

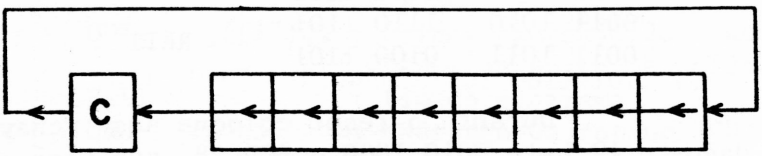
ENDEREÇO NA MEMÓRIA	CÓDIGO	(ASSEMBLER)
0000 0000 0000 0000	1110 1101	
0001	0100 0000	IN B, (C)
0010	1110 1101	
0011	0101 0110	IM 1
0100	0011 0001	
0101	1111 1111	LD SP, 2047
0110	0000 0111	
0111	1111 1011	- EI
1000	0111 0110	- HALT
1001	1000 0000	- ADD A,B
1010	0010 0111	- DAA
1011	1101 0011	
1100	0111 1111	OUT (127),A
1101	0111 0110	- HALT

ENDEREÇO NA MEMÓRIA	CÓDIGO	(ASSEMBLER)
0000 0000 0011 1000	1110 1101	IN A, (C)
0011 1001	0111 1000	
0011 1010	1110 1101	RETI
0011 1011	0100 1101	

Durante o reset coloque nas chaves de dados o 1º byte 0010 1000 = BCD 28, que será carregado em B após o reset. Enquanto o Z-80 se encontra em HALT aguardando interrupção assente o 2º byte 0011 1001 = BCD 39 e acione a chave de interrupção. Os dois bytes serão somados como se fossem binários, produzindo o resultado incorreto 0110 0001B que é corrigido para 0110 0111 = BCD 67 e gravado nos 8 LEDs. Faça novamente o reset e repita para novo par de BCD.

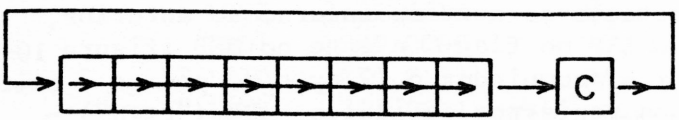
22 - INSTRUÇÕES DE ROTAÇÃO E DESLOCAMENTO

"RLr", código 1100 1011/0001 0..., roda o conteúdo do registro r uma casa para a esquerda, copiando o MSB na flag C e esta no LSB (figura 10-3); se r for o acumulador o 1º byte do código é omitido; para usar como operando o conteúdo do endereço indicado por um dos registros de índice - RL (rr+d) - o código é 11.1 1101/1100 1011/ dddd dddd/0001 0110. "RRr", código 1100 1011/0001 1..., faz a rotação no sentido inverso ao anterior, copiando o LSB no flag C e este no MSB (figura 10-4); se r for o acumulador o 1º byte do código é omitido e para os registros de índice - RR ("rr+d") - o código é 11.1 1101/1100 1011/dddd dddd/0001 1110.



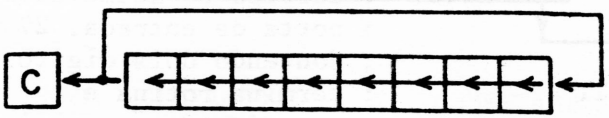
10-3 instrução "RLr"

"RLCr", código 1100 1011/0000 0..., gira 1 casa para a esquerda os bits de r, copiando o MSB no flag C e no LSB (figura 10-5); se r for o acumulador o 1º byte do código é omitido e para os registros de índice - RLC (rr+d) - o código é 11,1 1101/1100 1011/dddd dddd/0000 0110. "RRCr", código 1100 1011/0000 1..., executa a circulação no sentido inverso, copiando o LSB no flag C e no MSB (figura 10-6); se r é o acumulador o 1º byte do código é omitido e para os registros de índice - RRC (rr+d) - o código é 11,1 1101/1100 1011/ dddd dddd/0000 1110.

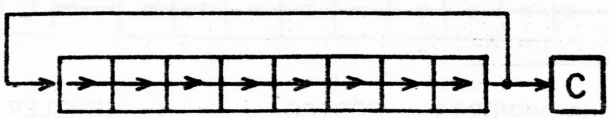


10-4 instrução "RRr"

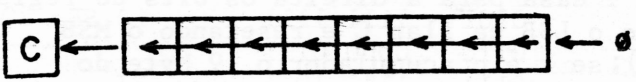
"SLAr", código 1100 1011/0010 0..., desloca 1 casa para a esquerda o conteúdo de r, copiando zero no LSB e o MSB na flag C (figura 10-7) se r é o acumulador o 1º byte do código é omitido e para os registros de índice - SLA (rr+d) - o código é 11.1 1101/1100 1011/dddd dddd/0010 0110.



10-5 instrução "RLCr"



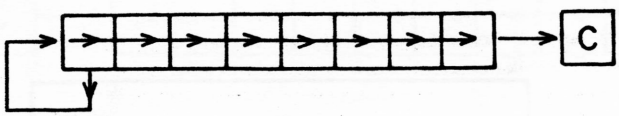
10-6 instrução "RRCr"



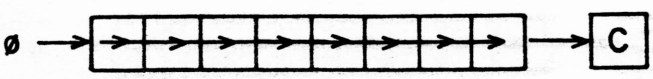
10-7 instrução "SLAr"

"SRA r", código 1100 1011/0010 1..., desloca 1 casa para a direita o conteúdo de r, co-

piando o LSB na flag C e mantendo o mesmo valor do MSB (figura 10-8); se r é o acumulador o 1º byte do código é omitido e para os registros de índice - SRA (rr+d) - o código é 11.1 1101/1100 1011/ dddd dddd/0010 1110.



10-8 instrução "SRAr"



10-9 instrução "SRLr"

"SRLr", código 1100 1011/0011 ...., desloca 1 casa para a direita os bits do registro r copiando o LSB na flag C e resetando o MSB (figura 10-9); se r é o acumulador o 1º byte do código é omitido e para os registros de índice - SRL (rr+d) - o código é 11.1 1101/1100 1011/dddd dddd/ 0010 1110.

No programa 25 um byte da porta de entrada é copiado no acumulador, é alterado pela instrução "SRL A" e o resultado colocado na porta de saída, habilitando-se interrupção modo 1. No en

dereço 56 é gravada a rotina de interrupção que coloca na pilha os conteúdos do par AF e retira-os para HL; o registro L agora contendo os bits de flag anteriormente em F é gravado na porta de saída.

PROGRAMA 25 - Deslocamento lógico à direita do acumulador

ASSEMBLER

```
LABEL   MNEM.   OPERANDO      COMENTÁRIOS
        ORG       0          ; origem da 1a. instrução
        LD      C,127        ; carrega 0111 1111B em C
        IN      A,(C)        ; carrega no acumulador by-
                             ; te da porta de entrada
        SRL     A            ; desloca byte do acumulador
                             ; para a direita, copiando o
                             ; LSB na flag C
BYTE    OUT     (C),A        ; grava na porta de saída by
                             ; te deslocado
        IM1                  ; seleciona interrupção modo
                             ; 1
        LD      SP,2047      ; inicia pilha em 2047
        EI                   ; habilita interrupção
        HALT                 ; parada até receber inter-
                             ; rupção
FLAG    ORG     56           ; origem da rotina FLAG que
                             ; apresenta nos 8 LEDs o con-
                             ; teúdo do registro F con-
                             ; tendo a flag C
        PUSH    AF           ; coloca AF na pilha, A em
                             ; 2045 e F em 2044
        POP     HL           ; retira de 2045=A para H e
                             ; de 2044=F para L
        OUT     (C),L        ; grava L=F na porta de saí-
                             ; da
        HALT                 ; fim da rotina FLAG
```

PROGRAMA 25

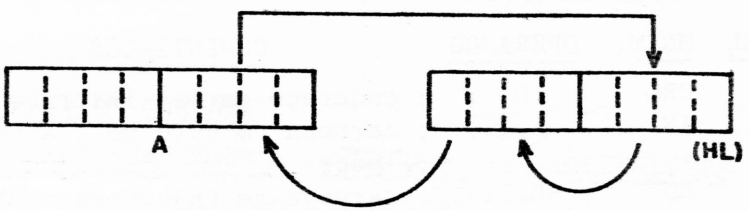
CÓDIGO DE MÁQUINA

POSIÇÃO NA MEMÓRIA	CÓDIGO	(ASSEMBLER)
0000 0000 0000 0000	0000 1110	LD C, 127
0001	0111 1111	
0010	1110 1101	IN A, (C)
0011	0111 1000	
0100	0011 1111	- SRL A
0101	1110 1101	OUT (C), A
0110	0111 1001	
0111	1110 1101	IM1
1000	0101 0110	
1001	0011 0001	
1010	1111 1111	LD SP, 2047
1011	0000 0111	
1100	1111 1011	- EI
1101	0111 0110	- HALT
0000 0000 0011 1000	1111 0101	- PUSH AF
0011 1001	1110 0001	- POP HL
0011 1010	1110 1101	OUT (C), L
0011 1011	0110 1001	
0011 1100	0111 0110	- HALT

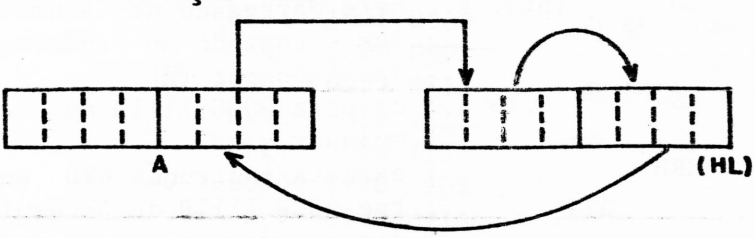
Durante o reset coloque um byte nas chaves de dados. Ao sair do reset o Z-80 irá gravá lo nos 8 LEDs, devidamente alterado pela instrução "SRL A" (confira pela figura 10-9),e permanecer em HALT aguardando interrupção, Acione a chave de interrupção, O conteúdo do registro L, que é o mesmo de F, será gravado nos 8 LEDs; verifique que o LSB do byte usado como teste é o mesmo do LED DO, correspondendo à flag C que é o LSB do registro F. Repita para outros bytes.

"RLD", código 1110 1101/0110 1111, transfere o 4 bits inferiores do conteúdo do endereço apontado por (HL) para os 4 bits superiores e estes para os 4 bits inferiores do acumulador, que são copiados nos 4 bits inferiores de (HL) (figura 10-10).

"RRD", código 1110 1101/0110 0111, transfere os 4 bits inferiores do conteúdo do endereço apontado por (HL) para os 4 bits inferiores do acumulador e estes para os 4 bits superiores de (HL), que são copiados nos 4 bits inferiores (figura 10-11).



10-10 instrução "RLD"



10-11 instrução "RRD"

No programa 26 um byte de porta de entrada é copiado no acumulador e dali transferido para o endereço 120 apontado por (HL). O acumulador é carregado com o byte 0000 1111 e executada a

instrução RRD, envolvendo o nybble 1111 do acumulador e o byte usado como teste. O conteúdo alterado do acumulador é gravado na porta de saída, selecionando-se interrupção modo 1 e habilitando-se interrupção. No endereço 56 de interrupção modo 1 a rotina BYTE carrega no acumulador o byte-teste já alterado apontado por (HL) e o grava na porta de saída.

PROGRAMA 26 - Rotação à direita de nybbles apontados por (HL)

ASSEMBLER

```
LABEL   MNEM.   OPERANDO        COMENTÁRIOS
        ORG     0           ; endereço da 1a. instrução
        IN      A,(C)       ; carrega um byte no acumu
                            ; lador
        LD      HL,128      ; carrega em HL endereço 1000
                            ; 0000B do byte a ser opera
                            ; do
        LD      (HL), A     ; byte carregado no acumula
                            ; dor é copiado no endereço
                            ; apontado por (HL)
        LD      A,15        ; carrega 0000 1111B no acu
                            ; mulador
        RRD                 ; executa instrução RRD en-
                            ; tre 0000 1111B do acumula
                            ; dor e byte copiado no ende
                            ; reço (HL) = 128
ACUM    OUT     (127),A     ; coloca na porta de saída
                            ; conteúdo do acumulador já
                            ; rodado
        IM1                 ; seleciona interrupção modo
                            ; 1
        LD      SP,2047     ; inicia pilha em 2047
        EI                  ; habilita interrupção
```

```
LABEL   MNEM.   OPERANDO        COMENTÁRIOS
        HALT                    ; parada até receber in-
                                ; terrupção
BYTE    ORG       56            ; endereço da rotina que
                                ; apresenta byte já rodado ,
                                ; gravada na posição de IM1.
        LD      A,(HL)          ; copia no acumulador conteú
                                ; do apontado por (HL), onde
                                ; está o byte já rodado
        OUT     (127),A         ; coloca acumulador na porta
                                ; de saída
        HALT                    ; fim da rotina BYTE
```

PROGRAMA 26

CÓDIGO DE MÁQUINA

ENDEREÇO NA MEMÓRIA	CÓDIGO	(ASSEMBLER)
0000 0000 0000 0000	1110 1101	
0001	0111 1000	IN A,(C)
0010	0010 0001	
0011	1000 0000	LD HL, 128
0100	0000 0000	
0101	0111 0111	- LD (HL), A
0110	0011 1110	
0111	0000 1111	LD A, 15
1000	1110 1101	
1001	0110 0111	RRD
1010	1101 0011	
1011	0111 1111	OUT (127),A
1100	1110 1101	
1101	0101 0110	IM1
1110	0011 0001	
1111	1111 1111	LD SP,2047
0000 0000 0001 0000	0000 0111	
1 0001	1111 1011	- EI
1 0010	0111 0110	- HALT

ENDEREÇO NA MEMÓRIA	CÓDIGO	(ASSEMBLER)
0000 0000 0011 1000	0111 1110	- LD A, (HL)
0011 1001	1101 0011	OUT (127),A
0011 1010	0111 1111	
0011 1011	0111 0110	- HALT

Durante o reset coloque o byte 1010 1100B nas chaves de dados, Ao sair do reset o Z-80 gravará "0000 1100" nos 8 LEDs, correspondendo ao conteúdo alterado do acumulador. Acione a chave de interrupção: a rotina BYTE será executada, gravando nos 8 LEDs "1111 1010", que é o resultado da rotação à direita do byte, Repita para outro byte, conferindo os resultados com a figura 10-11.

23 - SUB-ROTINAS

"CALL nn", código 1100 1101/ ..../.... ...., coloca na pilha o conteúdo do PC (contador do programa) e o carrega com nn, que passa a ser o endereço da sub-rotina a ser executada. "RET", código 1100 1001, retira da pilha o endereço seguinte à instrução "CALL nn" e o transfere para o PC, retornando à seqüência inicial.

"CALL cc, nn", código 11cc c100/.... ..../.... ...., é semelhante à instrução anterior mas a sub-rotina só é executada se a condição cc for verdadeira; cc assume a correspondência 000: NZ (Z=0), 001: Z(Z=1), 010: NC(C=0), 011:C(C=1) , 100: PO(P/V=0), 101: PE(P/V=1), 110:P(S=0), 111: M(S=1), "RET cc", código 11cc c000, retira da pilha o endereço seguinte à instrução "CALL" e o carrega no PC somente se a condição cc ocorrer, caso contrário continua o tratamento da sub-rotina.

No programa 27 a sub-rotina ECO gravada no endereço 1024, copia o byte da porta de entrada no acumulador e o coloca na porta de saída. Logo no início do programa a sub-rotina ECO é chamada e "ecoa" o 1º byte, em seguida habilitando-se interrupção modo 1. No endereço 56 da rotina de interrupção modo 1 é feita nova chamada para a sub rotina ECO, "ecoando" um segundo byte. Observe que a instrução "RET" no fim da sub-rotina retorna o programa para endereços diferentes a cada vez que é executada.

PROGRAMA 27 - Sub-rotina ECO

ASSEMBLER

```
LABEL   MNEM.   OPERANDO      COMENTÁRIOS
        ORG       0       ; endereço da 1a. instrução
        LD      SP,2047   ; inicia pilha em 2047
BYT1    CALL      ECO     ; chama sub-rotina ECO
        IM 1              ; seleciona interrupção modo
                          ; 1
        EI                ; habilita interrupção
        HALT              ; parada até ocorrer inter-
                          ; rupção
INT 1   ORG       56      ; endereço da rotina de in-
                          ; terrupção modo 1
BYT 2   CALL      ECO     ; chama sub-rotina ECO
        HALT              ; fim da rotina de interrup-
                          ; ção
ECO     ORG      1024     ; endereço da sub-rotina ECO
        IN      A,(255)   ; copia no acumulador byte
                          ; da porta de entrada
        OUT     (127),A   ; coloca acumulador na porta
                          ; de saída
        RET               ; fim da sub-rotina, retorna
```

```
                    ; à instrução seguinte àque
                    ; la em que ocorreu a cha
                    ; mada.
```

PROGRAMA 27

CÓDIGO DE MÁQUINA

ENDEREÇO NA MEMÓRIA	CÓDIGO	(ASSEMBLER)
0000 0000 0000 0000	0011 0001	
0001	1111 1111	LD SP, 2047
0010	0000 0111	
0011	1100 1101	
0100	0000 0000	CALL ECO
0101	0000 0100	
0110	1110 1101	IM 1
0111	0101 0110	
1000	1111 1011	- EI
1001	0111 0110	- HALT
0000 0000 0011 1000	1100 1101	
0011 1001	0000 0000	CALL ECO
0011 1010	0000 0100	
0011 1011	0111 0110	- HALT
0000 0100 0000 0000	1101 1010	IN A, (255)
0100 0000 0001	1111 1111	
0100 0000 0010	1101 0011	OUT (127),A
0100 0000 0011	0111 1111	
0100 0000 0100	1100 1001	- RET

Durante o reset coloque o 1º byte nas chaves de dados; ao sair do reset ele será ecoado pela chamada BYT1 e é feita a parada. Assente o 2º byte nas chaves de dados e acione a interrupção; a chamada BYT2 o ecoará.

24 - INSTRUÇÕES DE CONTROLE

Já vimos as instruções de controle "HALT" (parada), "RSTn" (reinício em n), "EI" e "DI" (habilitação de interrupção). "NOP", código 0000 0000, orienta ao microprocessador apenas incrementar + 1 o PC sem executar nenhuma operação.

CAPÍTULO 11
PROGRAMA MONITOR

Neste capítulo o leitor encontrará um processo mais adequado a entrada de programas extensos na memória. São apresentados ainda, nos seus traços principais, as teorias de fluxogramas e programas monitores. Ao final temos o esboço do jogo "batalha naval", como aplicação prática do que foi visto no capítulo anterior e neste.

1 - CARGA COM ECO DE PROGRAMAS EXTENSOS

O processo de carga de programas utilizando o chaveamento DMA é adequado para o estudo de instruções isoladas. Porém, quando o programa é extenso torna-se trabalhoso demais o chaveamento sucessivo de endereços. O programa abaixo é uma solução para o problema, permitindo o endereçamento automático além de fornecer eco da entrada (o byte recebido da porta de entrada é gravado na porta de saída).

No programa 28 o registro C é carregado com 127, apontando para a porta de saída, e o par HL carregado com 18, que é o primeiro endereço livre onde o programa extenso será gravado. No início da rotina REPT a pilha é iniciada em 2047, selecionado o modo 1 de interrupção e esta é habilitada, Inicia-se em seguida a rotina CARG que apresen-

ta o conteúdo de L, copia no acumulador o byte colocado na porta de entrada e o carrega no endereço apontado por (HL), incrementa + 1 o par HL, e finalmente fica em HALT até receber uma interrupção. No endereço 56 de tratamento de interrupção modo 1 é gravada a rotina ECO que coloca na porta de saída o byte carregado em (HL) e desvia incondicionalmente para a rotina REPT.

PROGRAMA 28 - Carga com eco de programas extensos

ASSEMBLER

```
LABEL   MNEM.   OPERANDO      COMENTÁRIOS
        ORG       0          ; endereço da 1a. instrução
        LD      C,127        ; carrega 0111 1111B em C
        LD      HL, 18       ; carrega em HL 1º endereço
                             ; disponível para o programa
                             ; extenso
REPT    LD      SP, 2047     ; inicia pilha em 2047
        IM1                  ; seleciona interrupção modo
                             ; 1
        EI                   ; habilita interrupção
CARG    OUT     (C), L       ; indica na porta de saída
                             ; byte inferior do endereço
                             ; que será carregado
        IN      A, (C)       ; carrega byte da porta de
                             ; entrada no acumulador
        LD      (HL), A      ; copia acumulador no ende-
                             ; reço (HL)
        INC      HL          ; incrementa +1 HL, apontan
                             ; do-o para o próximo ende
                             ; reço HL+1 a ser carregado
        HALT                 ; parada até ocorrer inter
                             ; rupção
ECO     ORG       56         ; endereço da rotina ECO que
```

LABEL	MNEM.	OPERANDO	COMENTÁRIOS
			; apresenta na porta de saí-
			; da o byte anterior grava
			; do no endereço cujo byte
			; inferior L foi apresentado
	OUT	(C), A	; coloca acumulador na porta
			; de saída
	JP	REPT	; volta para rotina REPT,
			; permanecendo neste looping
			; enquanto a interrupção es
			; tiver sendo solicitada.

PROGRAMA 28

CÓDIGO DE MÁQUINA

ENDEREÇO NA MEMÓRIA	CÓDIGO	(ASSEMBLER)
0000 0000 0000 0000	0000 1110	LD C, 127
0001	0111 1111	
0010	0010 0001	
0011	0001 0010	LD HL, 18
0100	0000 0000	
0101	0011 0001	
0110	1111 1111	LD SP, 2047
0111	0000 0111	
1000	1110 1101	IM 1
1001	0101 0110	
1010	1111 1011	- EI
1011	1110 1101	OUT (C), L
1100	0110 1001	
1101	1110 1101	IN A, (C)
1110	0111 1000	
1111	0111 0111	- LD (HL), A
0001 0000	0010 0011	- INC HL
0001 0001	0111 0110	- HALT

ENDEREÇO NA MEMÓRIA	CÓDIGO	(ASSEMBLER)
0000 0000 0011 1000	1110 1101	OUT (C), A
0011 1001	0111 1001	
0011 1010	1100 0011	
0011 1011	0000 0101	JP REPT
0011 1100	0000 0000	

Durante o reset coloque nas chaves de dados o byte de endereços 18 do programa extenso (note que ele não é o 1º byte do programa extenso, mas o 19º). Ao sair do reset o MICRO-GALENA gravará "0001 0010" nos 8 LEDs, correspondendo ao atual byte inferior do endereço em que este byte foi gravado. Enquanto o Z-80 permanece em HALT assente o próximo byte (endereço 19 do programa extenso) nas chaves de dados. Acione a chave de interrupção e deixe-a em $\overline{INT}$ = 0 (para baixo), conferindo nos 8 LEDs o byte anterior (endereço 18); neste ponto o Z-80 estará no looping de JP REPT - REPT - ECO - JP REPT, saltando a rotina CARG. Após a conferência desative a interrupção fazendo $\overline{INT}$ = 1 (chave para cima); o looping vai terminar por falta de pedido de interrupção e a rotina CARG será executada, colocando nos 8 LEDs o novo valor de L (0001 0011B = 19) e permanecendo em HALT até que a interrupção seja acionada, quando então a byte de endereço 19 será apresentado e o looping. Esta sequência poderá prosseguir indefinidamente, até preencher a capacidade da memória.

Ao chegar no endereço 56 = 0011 1000 B a rotina ECO deve ser mantida independente do que houver nessas posições no programa extenso. Portanto, os bytes dos endereços 56 a 60 do programa 28 são repetidos, sendo regravados.

Terminando de gravar o programa extenso solicita-se DMA e grava-se nos endereços zero a 18 e 56 a 60 os códigos corretos, cancelando o programa 28. Assim, o resultado final é o programa extenso sem nenhuma interferência do programa 28.

2 - FLUXOGRAMAS

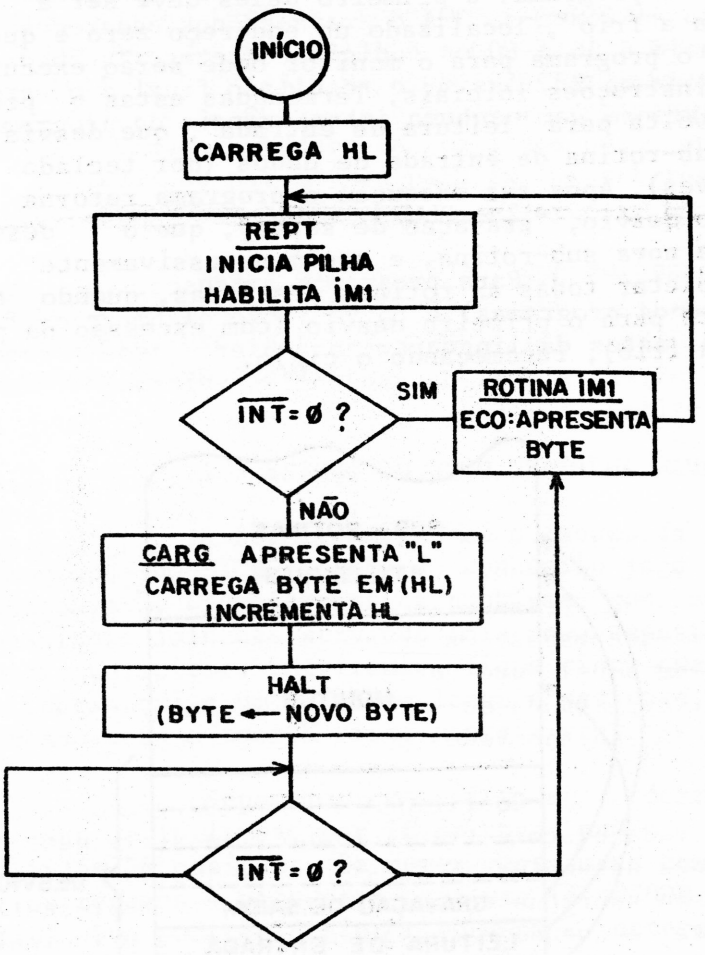
Para programas extensos (e mesmo pequenos programas) é conveniente elaborar o fluxograma antes da linguagem assembler. Isso permite uma análise global da seqüência, evitando erros de lógica.

Na figura 11-1 tomamos o fluxograma do programa 28 como exemplo. O início é colocado numa figura oval ou redonda e a partir daí o fluxo ocorre na seqüência vertical dos blocos, de cima para baixo. Retângulos indicam grupo de tarefas a serem executadas. Os losangos têm duas saídas e é seguida aquela que corresponder ao resultado do teste registrado no seu interior. Desvios da seqüência são apontados por setas.

Confirmado que o fluxograma está correto cada bloco é transformado numa série de instruções assembler. Rotinas que se repitam várias vezes são escritas como sub-rotinas, representadas aparte, e quando necessário o programa principal as chama.

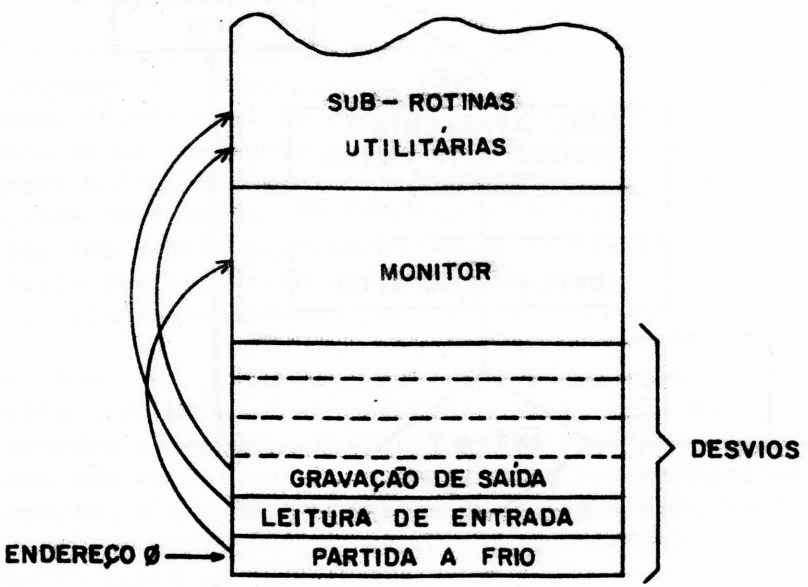
3 - ESQUEMA DE PROGRAMA MONITOR

Um esquema de programa é apresentado na figura 11-2, consistindo de três blocos: desvios, monitor e sub-rotinas utilitárias.



11-1 fluxograma do programa 28

O primeiro bloco é apenas uma série de desvios (tipo "JUMP" ou "CALL") para outros pontos do programa. O primeiro deles deve ser a "partida a frio", localizado no endereço zero e que desvia o programa para o monitor onde serão executadas as instruções iniciais. Terminadas estas o programa volta para "leitura de entrada", que desvia para a sub-rotina de entrada de dados (por teclado ou chaves). Após sua execução o programa retorna para novo desvio, "gravação de saída", que o desviará para nova sub-rotina, e assim sucessivamente até completar todas as rotinas desejadas, quando então volta para o primeiro desvio (com excessão da partida a frio), recomeçando o ciclo.



11-2 esquema de programa monitor

No bloco do monitor são colocadas instruções que carregam os valores iniciais de registros, gravam tabelas, reservam espaços na memória e rotinas que controlam a execução do programa. Os valores de tabelas e espaços reservados vão sendo alterados e preenchidos durante o processamento, e os resultados guardados em registros ou endereços, servindo de parâmetros para as sub-rotinas utilitárias.

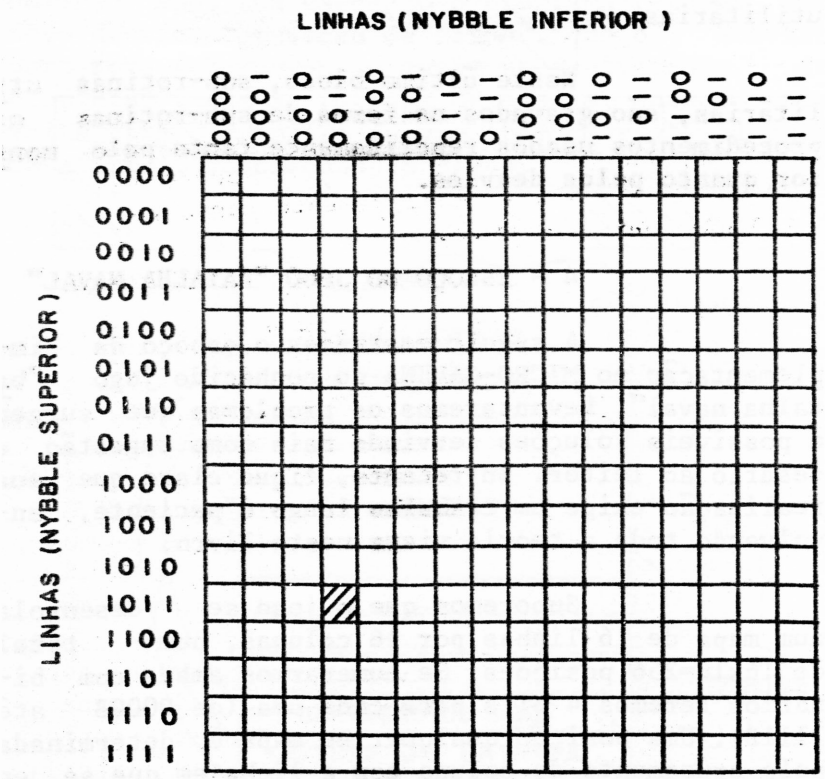
Neste último bloco, sub-rotinas utilitárias, são gravados na forma de sub-rotinas os procedimentos usados repetidamente tanto pelo monitor quanto pelos desvios.

4 - ESBOÇO DO JOGO "BATALHA NAVAL"

A seguir mostramos o esboço da implementação no MICRO-GALENA do conhecido jogo "batalha naval". Levantaremos os problemas que surgem e possíveis soluções servindo mais como sugestão e desafio ao leitor. Entretanto, fique claro que sua realização exige um trabalho longo e paciente, envolvendo toda a teoria vista neste livro.

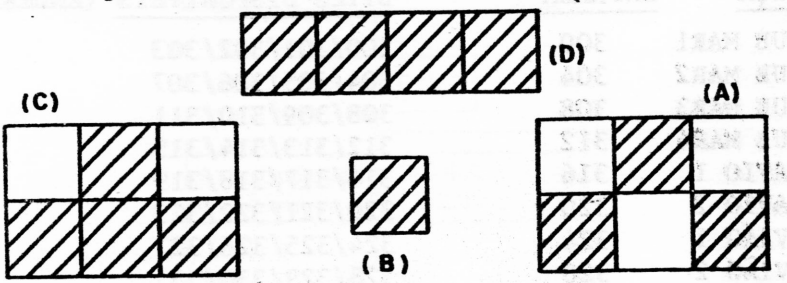
Suporemos que o jogo se desenrola num mapa de 16 linhas por 16 colunas, num total de 16x16=256 posições. Se numerarmos ambas com binários teremos 4 bits para cada uma (de 0000B até 1111B). Uma posição qualquer no mapa é determinada pelo cruzamento da coluna com a linha em que se encontra; por exemplo, na figura 11-3 a posição achurriada está na coluna 0011B e na linha 1011B. Convencionando escrever primeiro os 4 bits da coluna (nybble superior) e depois os 4 bits da linha (nybble inferior), formamos 1 byte que determina

a posição no mapa; no exemplo anterior o ponto achurriado está na posição 1011 0011B = 179.



11-3 mapa da "batalha naval"

O número e o formado das peças que compõem o jogo são de livre arbítrio do programador. Usaremos nesta explanação os modelos da figura 11-4, sendo 4 submarinos, 2 navios, 2 aviões e 1 encouraçado. Nesses modelos somente a parte achurriada pertence à peça; as posições em branco são vizinhanças no mapa.



11-4 "peças" do jogo
(a) avião (b) submarino
(c) navio (d) encouraçado

Para lançar no mapa suas 9 peças sem que o programador-adversário conheça as posições o computador deve gerar um número pseudo - randômico (pseudo-aleatório), o qual servirá de referência. Uma maneira prática e fácil é carregar no acumulador o registro de refresh ("LD A,R") realizar uma rotação para a esquerda ("RL A"). O resultado pode ser armazenado no monitor num endereço denominado PRM, ficando disponível como parâmetro da rotina de locação de peças. A primeira peça a ser lançada no mapa pode usar como endereço de locação o parâmetro visto, as outras devem mudá-lo a cada lançamento (por exemplo, usando a instrução "RL").

Uma tabela de locação de peças é guardada no monitor, com 9 entradas para as 9 peças.

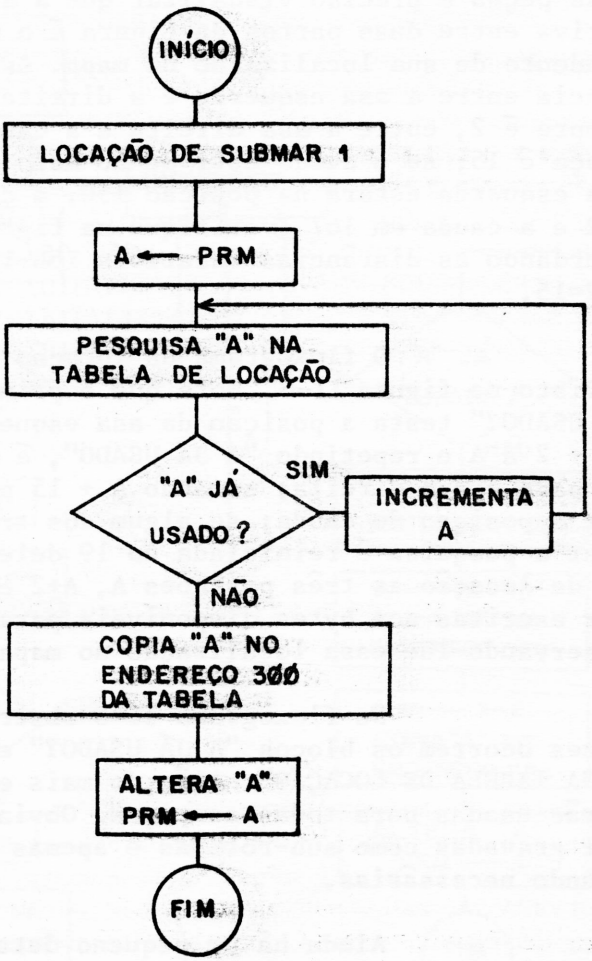
Para facilidade de programação cada peça tem 4 bytes à sua disposição. Abaixo um exemplo de tabela de locação:

TABELA DE LOCAÇÃO DE PEÇAS (NO MONITOR)

PEÇA	ENTRADA	BYTES DISPONÍVEIS (ENDER.)
SUB MAR1	300	300/301/302/303
SUB MAR2	304	304/305/306/307
SUB MAR3	308	308/309/310/311
SUB MAR4	312	312/313/314/315
NAVIO 1	316	316/317/318/319
NAVIO 2	320	320/321/322/323
AVIÃO 1	324	324/325/326/327
AVIÃO 2	328	328/329/330/331
ENCOURAÇADO	332	332/333/334/335

Na partida a frio esses 36 endereços (de 300 a 335) devem ser zerados.

A locação de submarinos é a mais fácil e seu fluxograma consta na figura 11-5.0 parâmetro PRM é copiado no acumulador e comparado com os bytes da tabela de locação (o par HL pode ser carregado com 300, BC com 36 e executada a instrução "CPIR"). Se for verificado que A=PRM está registrado na tabela significa que esta posição do mapa já está ocupada por outra peça; uma rotina deve incrementar o conteúdo de A=PRM e tentar novamente, Se A=PRM não consta na tabela então estará livre e é copiado num dos 4 bytes disponíveis para o submarino 1, reservando-lhe esta posição no mapa. O conteúdo de A é alterado e armazenado em PRM do monitor para referência de locação de outra peça.



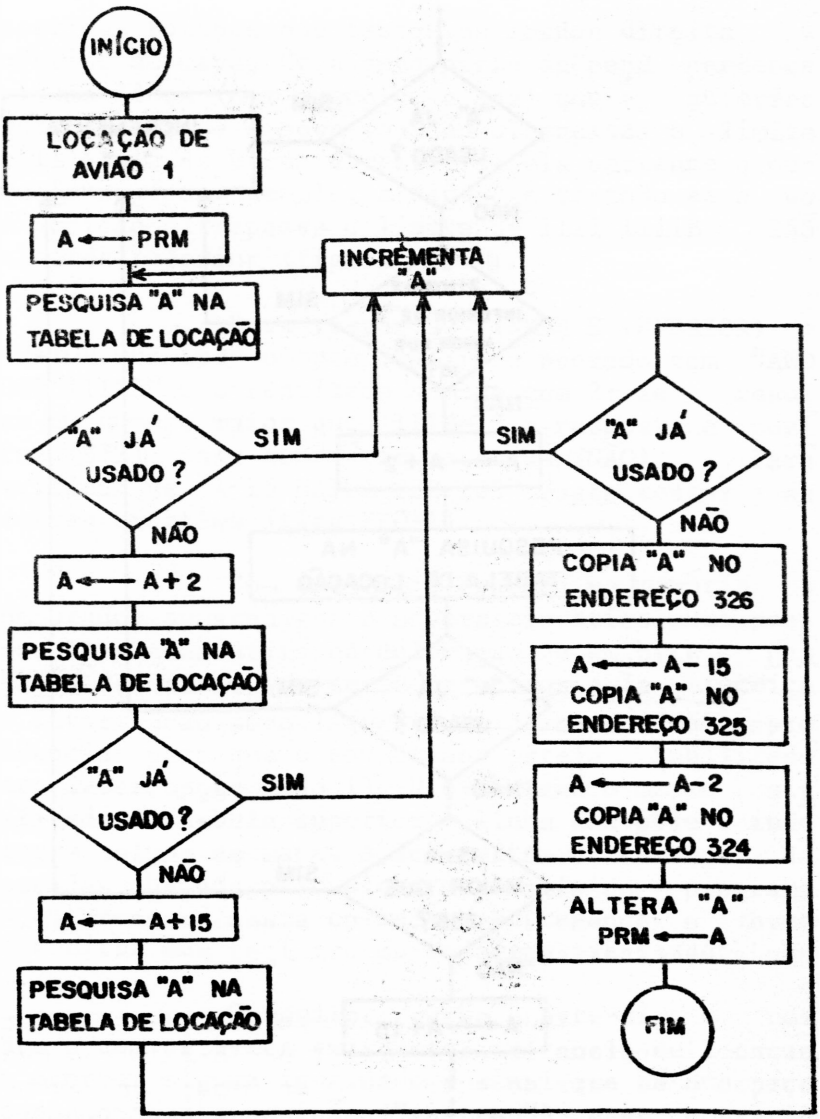
11-5 fluxograma da locação do submarino 1

Para o processo de locação das outras peças é preciso visualizar que a distância relativa entre duas partes da figura é o mesmo independente de sua localização no mapa. Assim, a distância entre a asa esquerda e a direita do avião sempre é 2, entre a asa direita e a cauda a diferença é 15; se o avião estiver no meio do mapa a asa esquerda estará na posição 150, a direita em 152 e a cauda em 167 (verifique na figura 11-3), guardando as distâncias relativas 152-150=2, 167-152=15.

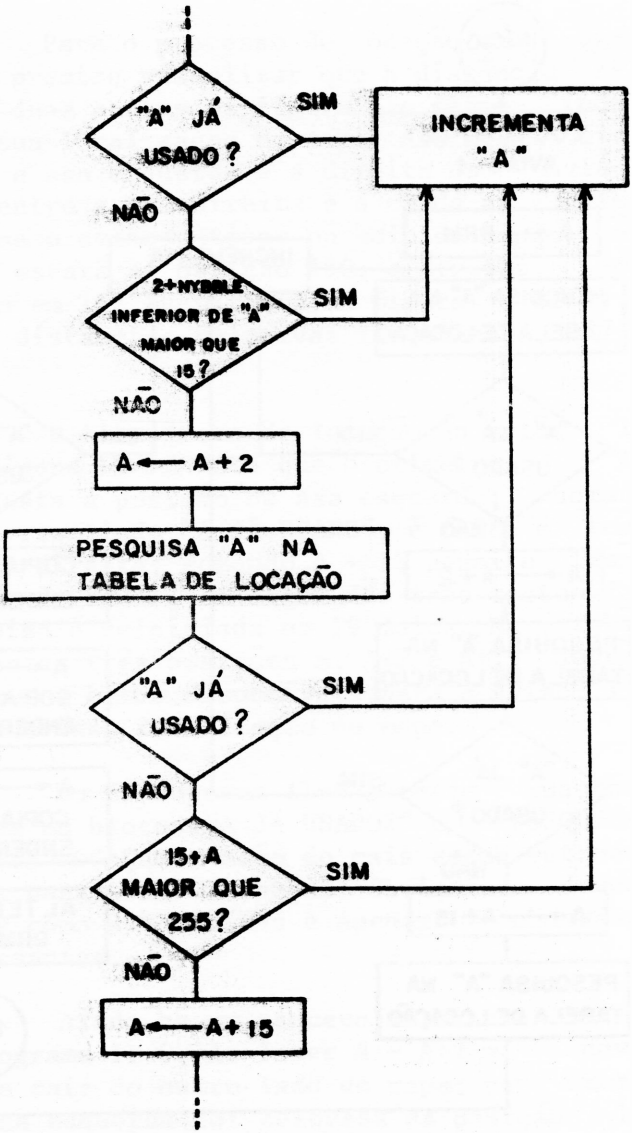
O fluxograma da locação do avião 1 é visto na figura 11-6. Note que o primeiro "A JÁ USADO?" testa a posição da asa esquerda; somando + 2 a A e repetindo "A JÁ USADO", é feito o teste para a asa direita; somando A + 15 permite testar a posição de cauda; se algum dos três testes falhar a pesquisa é reiniciada no 1º deles. Na tabela de locação as três posições A, A+2 e A+15 devem ser escritas nos bytes disponíveis para o avião 1, reservando-lhe essa localização no mapa.

No fluxograma 11-6 observe quantas vezes ocorrem os blocos "A JÁ USADO?" e "PESQUISA A NA TABELA DE LOCAÇÃO". Além do mais estas rotinas serão usadas para todas as peças. Obviamente devem ser gravadas como sub-rotinas e apenas chamadas quando necessárias.

Ainda há um pequeno detalhe a acertar no fluxograma 11-6. Ao fazer A - A+2 a nova posição pode cair do outro lado do mapa; por exemplo, se a asa esquerda foi colocada na posição 15 (canto direito superior) a asa direita cairia na posição 17 (canto esquerdo, uma linha mais embaixo), fragmentando a figura. Para evitar isso devem ser



11-6A fluxograma da locação do avião 1



11-6B – Complementação do fluxograma de locação do Avião 1

inseridas rotinas que testam as bordas direita e inferior do mapa. Se a nova parte da peça pertence à mesma linha (por exemplo, a asa) que a anterior basta testar se a nova posição ultrapassa o limite de 1111B=15 da borda direita; se ela pertence a outra linha (por exemplo, a cauda) é testado se a nova posição ultrapassa o limite de 1111 1111B= 255 da borda inferior (figura 11-68).

Para testar se "(NYBBLE INFERIOR) + 2 MAIOR QUE 15?" o byte inteiro é operado com "AND 0000 1111B" e o resultado somado com 2; se resultar um número maior que 1111B=15 a resposta é positiva (SIM), caso contrário negativa (NÃO). Para verificar se "A+15 MAIOR QUE 255" basta testar se ocorreu overflow (flag P/O=1).

Com o jogo já gravado na memória o computador aguarda que o adversário atire numa posição. Isso consistirá na colocação de um byte nas chaves de dados, que serão recebidos pela técnica de interrupção modo 1 (portanto o monitor deve providenciar para que o computador esteja habilitado para interrupção modo 1). O byte terá o mesmo significado de nybble superior = linha e nybble inferior = coluna do mapa. A sub-rotina de leitura de entradas chamada pelo bloco de desvio do programa se encarregará desta operação, entregando o byte ao monitor num registro ou endereço pré-fixado.

O monitor recebe o byte-tiro e compara-o com os bytes existentes na tabela de locação. Se ocorrer alguma igualdade é sinal que há uma peça ocupando aquela posição ("acertou"), caso contrário o tiro foi perdido (água). Este resultado é entregue à sub-rotina de saída de dados, que o apresenta

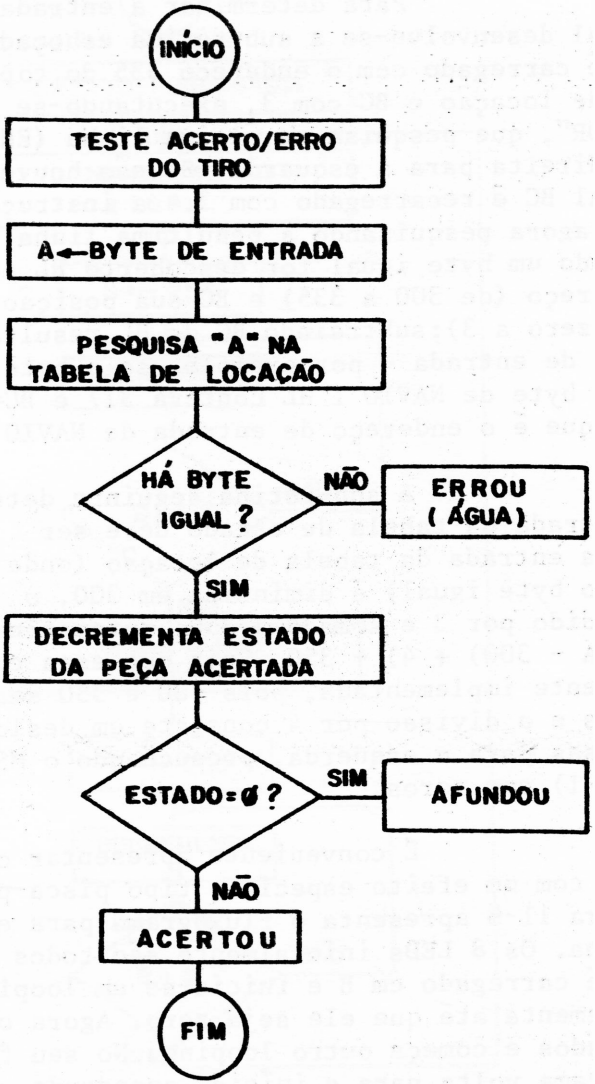
nos 8 LEDs; por exemplo, 0000 1111B para acerto e 1111 0000B para erro (ou 0000 0001B e 1111 1110 B).

Se houver acerto é necessário determinar se a peça foi parcialmente atingida ou totalmente ("afundou"). Para tanto o monitor inicialmente grava na memória outra tabela contendo o número total de posições ocupadas por cada peça, como abaixo:

TABELA DE ESTADO DAS PEÇAS

PEÇA	ENTRADA	ESTADO
SUB MAR 1	350	1
SUB MAR 2	351	1
SUB MAR 3	352	1
SUB MAR 4	353	1
NAVIO 1	354	4
NAVIO 2	355	4
AVIÃO 1	356	3
AVIÃO 2	357	3
ENCOURAÇADO	358	4

Ao encontrar na tabela de locação um byte igual a A o monitor verifica a qual entrada (de 300 a 335) ele pertence e aponta um ponteiro para a entrada (de 350 a 358) correspondente da tabela de estado das peças; o byte ali existente é decrementado - 1 e se o resultado for zero significa que a peça foi totalmente afundada. Observe que durante o jogo o estado da peça vai diminuindo a cada tiro que ela recebe. A figura 11-7 apresenta o fluxograma desta operação.

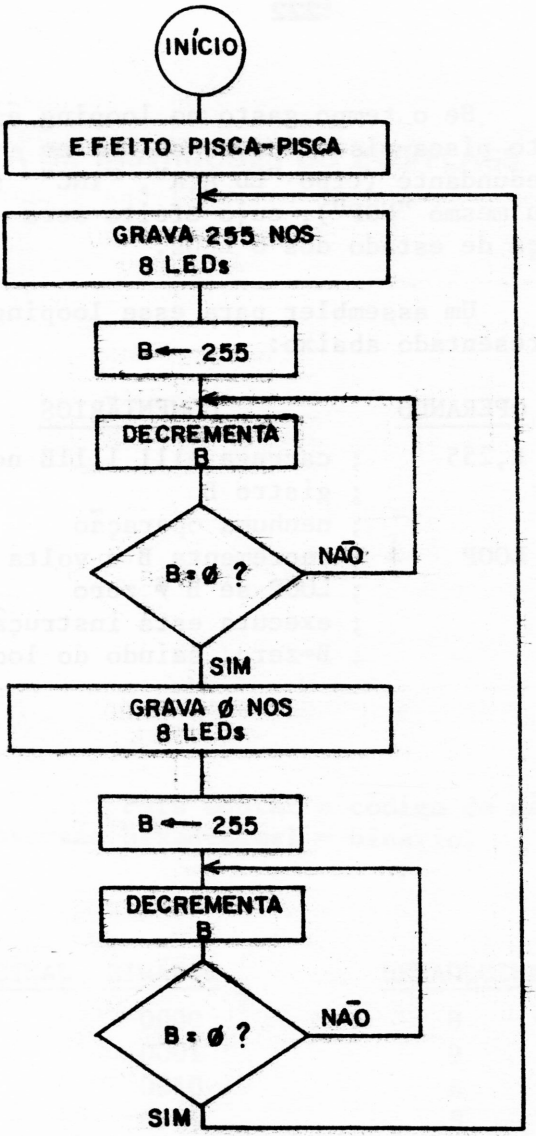


11-7 fluxograma do teste acerto/erro do tiro

Para determinar a entrada do byte igual desenvolve-se a sub-rotina esboçada a seguir. HL é carregado com o endereço 335 do topo da tabela de locação e BC com 3, executando-se a instrução "CPDR", que pesquisa na última linha (ENCOURAÇADO), da direita para a esquerda. Se não houver byte igual BC é recarregado com 3 e a instrução repetida, agora pesquisando a penúltima linha (AVIÃO 2). Quando um byte igual for descoberto HL conterá seu endereço (de 300 a 335) e BC sua posição na linha (de zero a 3); subtraindo BC de HL resultará o endereço de entrada - por exemplo, se o byte igual for o 2º byte de NAVIO 1 HL conterá 317 e BC=1, 317-1= 316 que é o endereço de entrada de NAVIO 1.

A sub-rotina seguinte determina qual a entrada da tabela de estado deve ser decrementada: a entrada da tabela de locação (onde foi descoberto byte igual) é diminuída em 300, o resultado dividido por 3 e somado a 350; matematicamente [(ENTRADA - 300) ÷ 4)] + 350. Veja que essa conta é facilmente implementada, pois 300 e 350 são números fixos e a divisão por 4 consiste em deslocar o byte 2 casas para a esquerda, preenchendo o MSB e o (MSB-1) com zeros.

É conveniente apresentar o "afundou" com um efeito especial, tipo pisca-pisca. A figura 11-8 apresenta o fluxograma para essa sub-rotina. Os 8 LEDs inicialmente são todos acesos, 255 é carregado em B e inicia-se um looping que o decrementa até que ele seja zero. Agora os LEDs são apagados e começa outro looping. No seu final o programa volta para o início, acendendo novamente os LEDs. Durante o tempo gasto no 1º looping os LEDs estavam acesos, no tempo gasto no 2º looping apagados, repetindo-se o ciclo indefinidamente.



11-8 fluxograma do efeito pisca-pisca

Se o tempo gasto no looping é pouco para o efeito pisca-pisca insere-se uma ou mais instrução redundante (tipo "LD A,A", "INC" seguido de "DEC", ou mesmo "NOP"), cujo efeito será atrasar a mudança de estado dos 8 LEDs.

Um assembler para esse looping de atraso é apresentado abaixo:

```
LABEL  MNEM.  OPERANDO      COMENTÁRIOS
ATRS   LD     B,255        ; carrega 1111 1111B no re-
                            ; gistro B
LOOP   NOP                  ; nenhuma operação
       DJNZ   LOOP         ; decrementa B e volta para
                            ; LOOP se B ≠ zero
PRÓX                        ; executa esta instrução se
                            ; B=zero, saindo do looping
```

APÊNDICE
LISTA DE INSTRUÇÕES DO Z-80/FLAGS

Para montar o código de máquina faça a conversão hexadecimal - binário.

HEXADECIMAL	BINÁRIO	HEXADECIMAL	BINÁRIO
0	0000	8	1000
1	0001	9	1001
2	0010	A	1010
3	0011	B	1011
4	0100	C	1100
5	0101	D	1101
6	0110	E	1110
7	0111	F	1110

INSTRUÇÕES DE TRANSFERÊNCIA DE 8 BITS

	A	B	C	D	E	H	L	(HL)	(BC)	(DE)	(nn)	n
LD A,.	7F	78	79	7A	7B	7C	7D	7E	0A	1A	3AXXXX	3EXX
LD B,.	47	40	41	42	43	44	45	46				06XX
LD C,.	4F	48	49	4A	4B	4C	4D	4E				0EXX
LD D,.	57	50	51	52	53	54	55	56				16XX
LD E,.	5F	58	59	5A	5B	5C	5D	5E				1EXX
LD H,.	67	60	61	62	63	64	65	66				26XX
LD L,.	6F	68	69	6A	6B	6C	6D	6E				2EXX
LD (HL),.	77	70	71	72	73	74	75					36XX
LD (BC),.	02											
LD (DE),.	12											
LD (nn),.	32XXXX											

	A	B	C	D	E	H	L
LD.,(IX+d)	DD7EXX	DD46XX	DD4EXX	DD56XX	DD5EXX	DD66XX	DD6EXX
LD.,(IY+d)	FD7EXX	FD46XX	FD4EXX	FD56XX	FD5EXX	FD66XX	FD6EXX
LD(IX+d),.	DD77XX	DD70XX	DD71XX	DD72XX	DD73XX	DD74XX	DD75XX
LD(IY+d),.	FD77XX	FD70XX	FD71XX	FD72XX	FD73XX	FD74XX	FD75XX

LD(IX+d),n DD36XXXX LD(IY+d),n FD36XXXX

		S	Z	H	P/V	N	C	
IN A,(n)	DBXX	-	-	-	-	-	-	
OUT(n),A	D3XX	-	-	-	-	-	-	
INI	EDA2	?	*	?	?	1	-	IN(HL),(C);INC HL, DEC B
INIR	ED82	?	1	?	?	1	-	COMO INI,REPETIR ATÉ B=0
IND	EDAA	?	*	?	?	1	-	IN(HL),(C); DEC HL, DEC B
INDR	EDBA	?	1	?	?	1	-	COMO IND, REPETIR ATÉ B=0
OUTI	EDA3	?	*	?	?	1	-	OUT(C), (HL), INC HL, DEC B
OTIR	EDB3	?	1	?	?	1	-	COMO OUTI, REPETIR ATÉ B=0
OUTD	EDAB	?	*	?	?	1	-	OUT(C),(HL); DEC HL; DEC B
OTDR	EDBB	?	1	?	?	1	-	COMO OUTD, REPETIR ATÉ B=0

INSTRUÇÕES LÓGICAS E ARITMÉTICAS DE 8 BITS

	B	C	D	E	H	L	(HL)	A	n	(IX+d)	(IY+d)	S	Z	H	P/V	N	C
ADD .	80	81	82	83	84	85	86	87	C6XX	DD86XX	FD86XX	*	*	*	*	0	*
ADC .	88	89	8A	8B	8C	8D	8E	8F	CEXX	DD8EXX	FD8EXX	*	*	*	*	0	*
SUB .	90	91	92	93	94	95	96	97	D6XX	DD96XX	FD96XX	*	*	*	*	1	*
SBC .	98	99	9A	9B	9C	9D	9E	9F	DEXX	DD9EXX	FD9EXX	*	*	*	*	1	*
AND .	A0	A1	A2	A3	A4	A5	A6	A7	E6XX	DDA6XX	FDA6XX	*	*	1	*	0	0
XOR .	A8	A9	AA	AB	AC	AD	AE	AF	EEXX	DDAEXX	FDAEXX	*	*	0	*	0	0
OR .	B0	B1	B2	B3	B4	B5	B6	B7	F6XX	DDB6XX	FDB6XX	*	*	0	*	0	0
CP .	B8	B9	BA	BB	BC	BD	BE	BF	FEXX	DDBEXX	FDBEXX	*	*	*	*	1	*
INC .	04	0C	14	1C	24	2C	34	3C		DD34XX	FD34XX	*	*	*	*	0	-
DEC .	05	0D	15	1D	25	2D	35	3D		DD35XX	FD35XX	*	*	*	*	1	-

		S	Z	H	P/V	N	C	
DAA	27	*	*	*	*	-	*	AJUSTE DE RESULTADO BCD NO AC
CPL	2F	-	-	1	-	1	-	COMPLEMENTO A UM AO AC
NEG	ED44	*	*	*	*	1	*	COMPLEMENTO A DOIS DO AC

		S	Z	H	P/V	N	C
LD A,I	ED57	*	*	0	.	0	-
LD A,R	ED5F	*	*	0	.	0	-
LD I,A	ED47	-	-	-	-	-	-
LD R,A	ED4F	-	-	-	-	-	-

INSTRUÇÕES DE TRANSFERÊNCIA E PERMUTA DE 16 BITS

	BC	DE	HL	SP	IX	IY
LD.., nn	01XXXX	11XXXX	21XXXX	31XXXX	DD21XXXX	FD21XXXX
LD..,(nn)	ED4BXXXX	ED5BXXXX	2AXXXX	ED7BXXXX	DD2AXXXX	FD2AXXXX
LD(nn),,,	ED43XXXX	ED53XXXX	22XXXX	ED73XXXX	DD22XXXX	FD22XXXX
LD SP,..			F9		DDF9	FDF9

	BC	DE	HL	AF	IX	IY
PUSH..	C5	D5	E5	F5	DDE5	FDE5
POP..	C1	D1	E1	F1	DDE1	FDE1

EX (SP), HL	E3	EX DE,HL	EB	
EX (SP), IX	DDE3	EX AF,AF'	08	
EX (SP), IY	FDE3	EXX	D9	BC-BC' DE-DE' HL-HL'

INSTRUÇÕES DE TRANSFERÊNCIA DE BLOCOS DE PESQUISA

		S	Z	H	P/V	N	C	
LDI	EDA0	-	-	0	*	0	-	LD(DE),(HL);INC HL; INC DE; DEC BC
LDIR	EDB0	-	-	0	0	0	-	COMO LDI,REPETIR ATÉ BC = 0
LDD	EDA8	-	-	0	*	0	-	LD(DE),(HL);DEC HL; DEC DE; DEC BC
LDDR	EDB8	-	-	0	0	0	-	COMO LDD, REPETIR ATÉ BC=0
CPI	EDA1	*	*	*	*	1	-	CP(HL); INC HL; DEC BC
CPIR	EDB1	*	*	*	*	1	-	COMO CPI, REPETIR ATÉ BC = 0 ou A = (HL)
CPD	EDA9	*	*	*	*	1	-	CP(HL); DEC HL; DEC BC
CPDR	EDB9	*	*	*	*	1	-	COMO CPD, REPETIR ATÉ BC =0 ou A = (HL)

INSTRUÇÕES ARITMÉTICAS DE 16 BITS

	BC	DE	HL	SP	IX	IY	S	Z	H	P/V	N	C
INC	03	13	23	33	DD23	FD23	-	-	-	-	-	-
DEC	0B	1B	2B	3B	DD2B	FD2B	-	-	-	-	-	-
ADD HL,.	09	19	29	39			-	-	?	-	0	*
ADC HL,.	ED4A	ED5A	ED6A	ED7A			*	*	?	*	0	*
SBC HL,.	ED42	ED52	ED62	ED72			*	*	?	*	1	*
ADD IX,.	DD09	DD19		DD39	DD29		-	-	?	-	0	*
ADD IY,.	FD09	FD19		FD39		FD29	-	-	?	-	0	*

INSTRUÇÕES DE DESLOCAMENTO E ROTAÇÃO

	B	C	D	E	H	L	(HL)	A	(IX+d)	(IY+d)
RR	CB18	CB19	CB1A	CB1B	CB1C	CB1D	CB1E	1F	DDCBXX1E	FDCBXX1E
RL	CB10	CB11	CB12	CB13	CB14	CB15	CB16	17	DDCBXX16	FDCBXX16
RRC	CB08	CB09	CB0A	CB0B	CB0C	CB0D	CB0E	0F	DDCBXX0E	FDCBXX0E
RLC	CB00	CB01	CB02	CB03	CB04	CB05	CB06	07	DDCBXX06	FDCBXX06
SRA	CB28	CB29	CB2A	CB2B	CB2C	CB2D	CB2E	2F	DDCBXX2E	FDCBXX2E
SLA	CB20	CB21	CB22	CB23	CB24	CB25	CB26	27	DDCBXX26	FDCBXX26
SRL	CB38	CB39	CB3A	CB3B	CB3C	CB3D	CB3E	3F	DDCBXX3E	FDCBXX3E

	S	Z	H	P/V	N	C	
RR/RL	*	*	0	*	0	*	DESLOC. CIRC. Ã DIR/ESQ.ATRAV. CARRY
RRC/RLC	*	*	0	*	0	*	DESLOC. CIRC. Ã DIR/ESQ.
SRA/SLA	*	*	0	*	0	*	DESLOC. ARITMÉTICO Ã DIR./ESQ.
SRL	*	*	0	*	0	*	DESLOC. LÓGICO Ã DIR./ESQ.

		S	Z	H	P/V	N	C	
RRCA	OF	-	-	0	-	0	*	DESLOC. CIRC. DO AC Ã DIR.
RLCA	07	-	-	0	-	0	*	DESLOC. CIRC. DO AC Ã ESQ.
RRA	1F	-	-	0	-	0	*	DESLOC. CIRC. DO AC Ã DIR. ATRAVÉS DO CARRY
RLA	17	-	-	0	-	0	*	DESLOC. CIRC. DO AC Ã ESQ. ATRAVÉS DO CARRY
RLD	ED6F	*	*	0	*	0	-	DESLOC. CIRC. DE DÍGITO Ã ESQ. ATRAV. AC E (HL)
RRD	ED67	*	*	0	*	0	-	DESLOC. CIRC. DE DÍGITO Ã DIR. ATRAV. AC E (HL)

"JUMP'S", "CALL'S" e "RETURN'S"

	Z	NZ	C	NC	PE	PO	M	P
JP	CAXXXX	C2XXXX	DAXXXX	D2XXXX	EAXXXX	E2XXXX	FAXXXX	F2XXXX
CALL	CCXXXX	C4XXXX	DCXXXX	D4XXXX	ECXXXX	E4XXXX	FCXXXX	F4XXXX
RET	C8	CO	D8	DO	E8	EO	F8	FO
JR	28XX	20XX	38XX	3OXX				

	INCONDICIONAIS	(HL)	(IX)	(IY)
JP	C3XXXX	E9	DDE9	FDE9
CALL	CDXXXX			
RET	C9			
JR	18XX			

RST	00	08	16	24	32	40	48	56
	C7	CF	D7	DF	E7	EF	F7	FF

DJNZ	10XX	DEC B; JR NZ,d
RETI	ED4D	RETORNO DE INTERRUPÇÃO
RETN	ED45	RETORNO DE INTERRUPÇÃO NÃO MASCARADA

INSTRUÇÕES DE CONTROLE

		S	Z	H	P/V	N	C	
NOP	00	-	-	-	-	-	-	INSTRUÇÃO DESTITUÍDA DE OPERAÇÃO
HALT	76	-	-	-	-	-	-	
CCF	3F	-	-	?	-	0	*	COMPLEMENTA O BIT CARRY
SCF	37	-	-	0	-	0	1	SETA O BIT CARRY
EI	FB	-	-	-	-	-	-	HABILITA INTERRUPÇÃO
DI	F3	-	-	-	-	-	-	INIBE INTERRUPÇÃO
IM 0	ED46	-	-	-	-	-	-	INTERRUPÇÃO MODO - 0
IM 1	ED56	-	-	-	-	-	-	INTERRUPÇÃO MODO - 1
IM 2	ED5E	-	-	-	-	-	-	INTERRUPÇÃO MODO - 2

INSTRUÇÕES DE ENTRADA E SAÍDA

	A	B	C	D	E	H	L	S	Z	H	P/V	N	C
IN., (C)	ED78	ED40	ED48	ED50	ED58	ED60	ED68	*	*	*	*	0	-
OUT (C),.	ED79	ED41	ED49	ED51	ED59	ED61	ED69	-	-	-	-	-	-

INSTRUÇÕES DE MANIPULAÇÃO E TESTE DE BITS

	B	C	D	E	H	L	(HL)	A	(IX+d)	(IY+d)
BIT 0	CB40	CB41	CB42	CB43	CB44	CB45	CB46	CB47	DDCBXX46	FDCBXX46
BIT 1	CB48	CB49	CB4A	CB4B	CB4C	CB4D	CB4E	CB4F	DDCBXX4E	FDCBXX4E
BIT 2	CB50	CB51	CB52	CB53	CB54	CB55	CB56	CB57	DDCBXX56	FDCBXX56
BIT 3	CB58	CB59	CB5A	CB5B	CB5C	CB5D	CB5E	CB5F	DDCBXX5E	FDCBXX5E
BIT 4	CB60	CB61	CB62	CB63	CB64	CB65	CB66	CB67	DDCBXX66	FDCBXX66
BIT 5	CB68	CB69	CB6A	CB6B	CB6C	CB6D	CB6E	CB6F	DDCBXX6E	FDCBXX6E
BIT 6	CB70	CB71	CB72	CB73	CB74	CB75	CB76	CB77	DDCBXX76	FDCBXX76
BIT 7	CB78	CB79	CB7A	CB7B	CB7C	CB7D	CB7E	CB7F	DDCBXX7E	FDCBXX7E
RES 0	CB80	CB81	CB82	CB83	CB84	CB85	CB86	CB87	DDCBXX86	FDCBXX86
RES 1	CB88	CB89	CB8A	CB8B	CB8C	CB8D	CB8E	CB8F	DDCBXX8E	FDCBXX8E
RES 2	CB90	CB91	CB92	CB93	CB94	CB95	CB96	CB97	DDCBXX96	FDCBXX96
RES 3	CB98	CB99	CB9A	CB9B	CB9C	CB9D	CB9E	CB9F	DDCBXX9E	FDCBXX9E
RES 4	CBA0	CBA1	CBA2	CBA3	CBA4	CBA5	CBA6	CBA7	DDCBXXA6	FDCBXXA6
RES 5	CBA8	CBA9	CBAA	CBAB	CBAC	CBAD	CBAE	CBAF	DDCBXXAE	FDCBXXAE
RES 6	CBB0	CBB1	CBB2	CBB3	CBB4	CBB5	CBB6	CBB7	DDCBXXB6	FDCBXXB6
RES 7	CBB8	CBB9	CBBA	CBBB	CBBC	CBBD	CBBE	CBBF	DDCBXXBE	FDCBXXBE
SET 0	CBC0	CBC1	CBC2	CBC3	CBC4	CBC5	CBC6	CBC7	DDCBXXC6	FDCBXXC6
SET 1	CBC8	CBC9	CBCA	CBCB	CBCC	CBCD	CBCE	CBCF	DDCBXXCE	FDCBXXCE
SET 2	CBD0	CBD1	CBD2	CBD3	CBD4	CBD5	CBD6	CBD7	DDCBXXD6	FDCBXXD6
SET 3	CBD8	CBD9	CBDA	CBDB	CBDC	CBDD	CBDE	CBDF	DDCBXXDE	FDCBXXDE
SET 4	CBE0	CBE1	CBE2	CBE3	CBE4	CBE5	CBE6	CBE7	DDCBXXE6	FDCBXXE6
SET 5	CBE8	CBE9	CBEA	CBEB	CBEC	CBED	CBEE	CBEF	DDCBXXEE	FDCBXXEE
SET 6	CBF0	CBF1	CBF2	CBF3	CBF4	CBF5	CBF6	CBF7	DDCBXXF6	FDCBXXF6
SET 7	CBF8	CBF9	CBFA	CBFB	CBFC	CBFD	CBFE	CBFF	DDCBXXFE	FDCBXXFE

FLAGS DE CONDIÇÃO

MNE	OP	FLAGS			
		C	Z	S	P/V
.SET	b,s	-	-	-	-
.RES	b,s	-	-	-	-
.LD	r,s	-	-	-	-
.LD	s,r	-	-	-	-
.LD	s,n	-	-	-	-
.LD	A,s	-	-	-	-
.LD	s,A	-	-	-	-
.LDI		-	x	x	.
.LDIR		-	x	x	0
.LDD		-	x	x	.
.LDDR		-	x	x	.
.IN	A,(n)	-	-	-	-
.IN	r,(C)	-	.	.	.
.OUT	(n),A	-	-	-	-
.OUT	(C),r	-	-	-	-
.INI		-	.	x	x
.INIR		-	1	x	x

.IND		-	.	x	x
.INDR		-	1	x	x
.OUTI		-	.	x	x
.OTIR		-	1	x	x
.OUTD		-	.	x	x
.OTDR		-	1	x	x
.INC	s	-	.	.	.
.DEC	s	-	.	.	.
.CPL		-	-	-	-
.NEG		.	.	.	.
.ADD	s	.	.	.	.
.ADC	s	.	.	.	.
.SUB	s	.	.	.	.
.SBC	s	.	.	.	.
.DAA		.	.	.	.
.AND	s	0	.	.	.
.XOR	s	0	.	.	.
.OR	s	0	.	.	.
.BIT	b,s	-	.	x	x
.CP	s	.	.	.	.
.CPI	s	-	.	x	.
.CPIR		-	.	x	.
.CPD		-	.	x	.
.CPDR		-	.	x	.

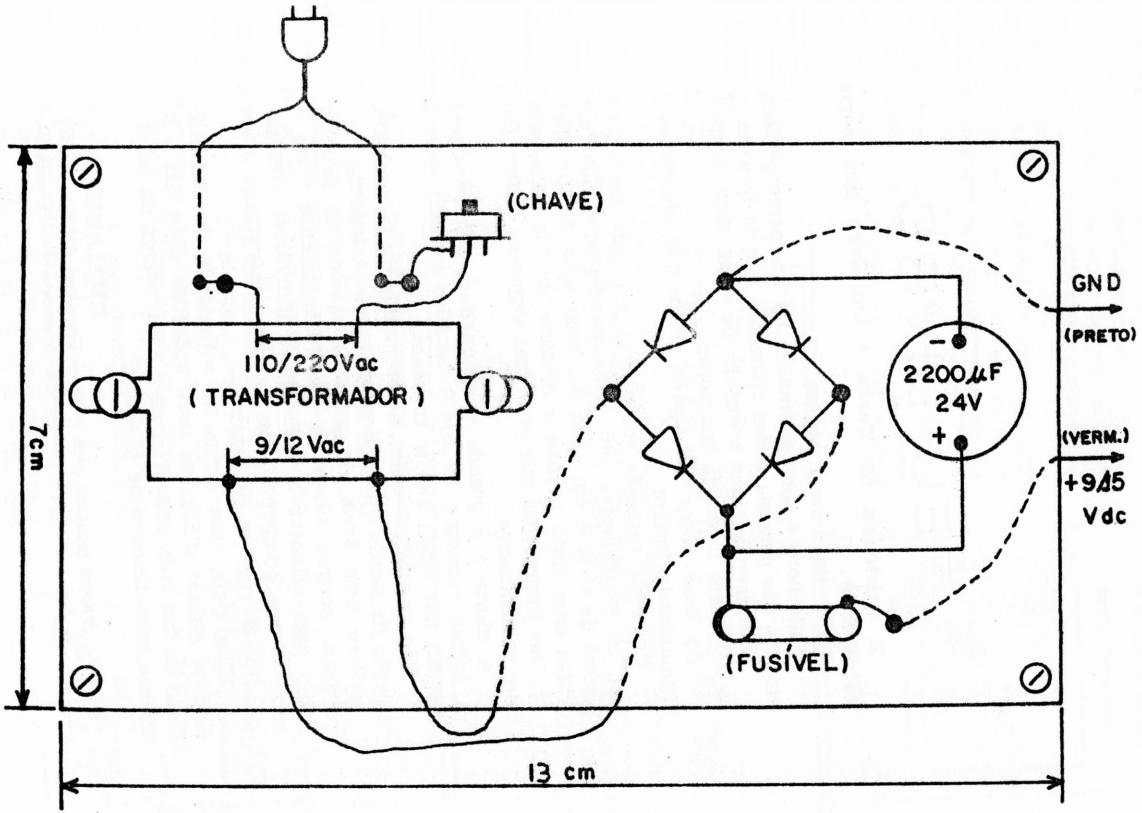
.RLC	s	.	.	.	.
.RRC	s	.	.	.	.
.RL	s	.	.	.	.
.RR	s	.	.	.	.
.SLA	s	.	.	.	.
.SRA	s	.	.	.	.
.SRL	s	.	.	.	.
.RLD		-	.	.	.
.RRD		-	.	.	.
.LD	ss,nn	-	-	-	-
.LD	ss,(nn)	-	-	-	-
.LD	(nn),ss	-	-	-	-
.LD	SP,ss	-	-	-	-
.PUSH	ss	-	-	-	-
.POP	ss	-	-	-	-
.EX	DE,HL	-	-	-	-
.EX	AF,AF'	-	-	-	-
.EXX		-	-	-	-
.EX	(SP),ss	-	-	-	-
.INC	ss	-	-	-	-
.DEC	ss	-	-	-	-
.ADD	HL,ss	.	-	-	-
.ADD	IX,ss	.	-	-	-
.ADD	IY,ss	.	-	-	-

.ADC	HL,ss	.	.	.	.
.SBC	HL,ss	.	.	.	.
.SCF		1	-	-	-
.CCF		.	-	-	-
.EI		-	-	-	-
.DI		-	-	-	-
.IM	0	-	-	-	-
.IM	1	-	-	-	-
.IM	2	-	-	-	-
.HALT		-	-	-	-
.NOP		-	-	-	-
.JP	nn	-	-	-	-
.JR	e	-	-	-	-
.JP	cc,nn	-	-	-	-
.JR	cc,e	-	-	-	-
.JP	(HL)	-	-	-	-
.JP	(ss)	-	-	-	-
.DJNZ	e	-	-	-	-
.CALL	nn	-	-	-	-
.RST	p	-	-	-	-
.CALL	cc,nn	-	-	-	-
.RET		-	-	-	-
.RET	cc	-	-	-	-

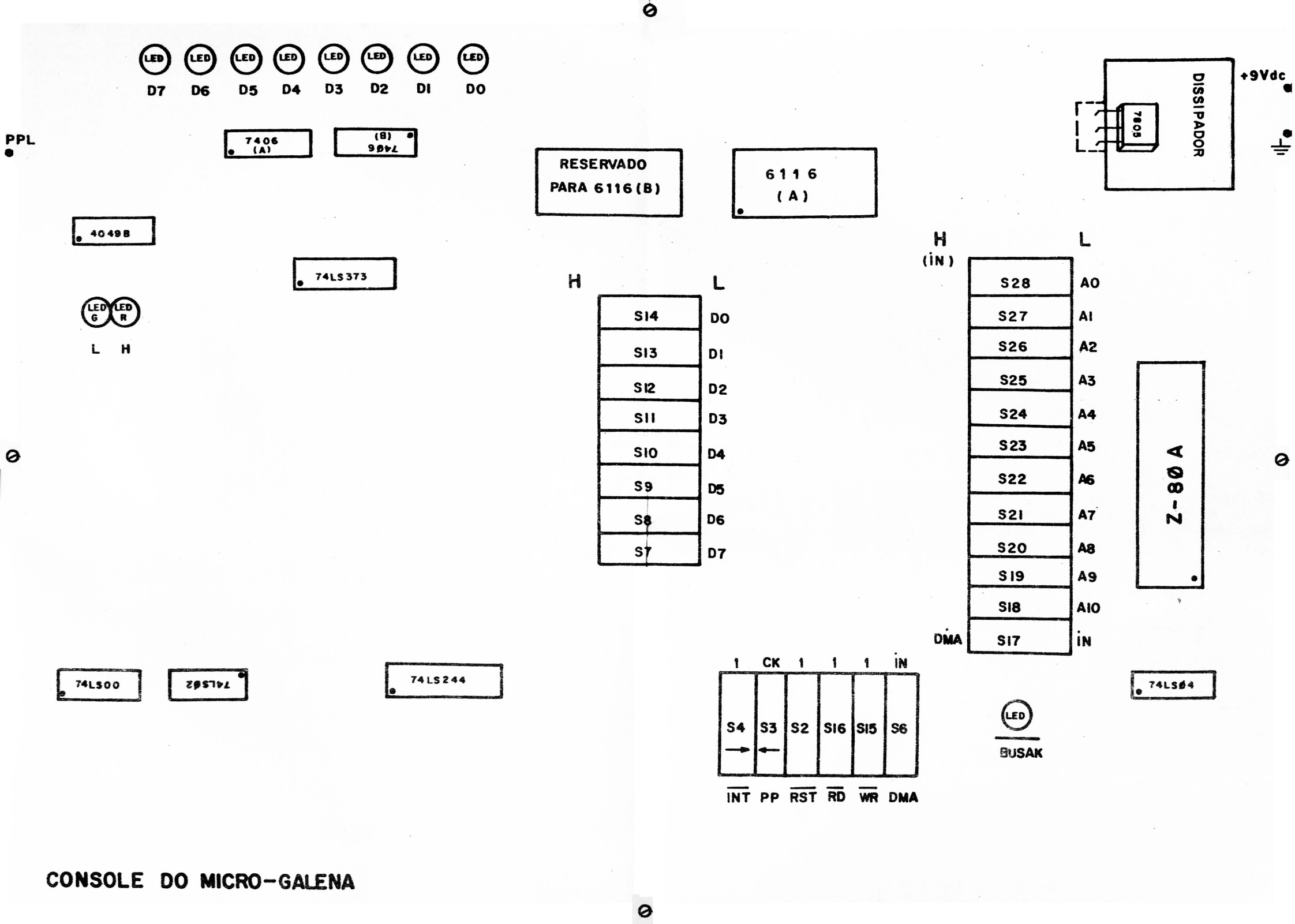
.RETI	-	-	-	-	
.RETN	-	-	-	-	

FLAGS:

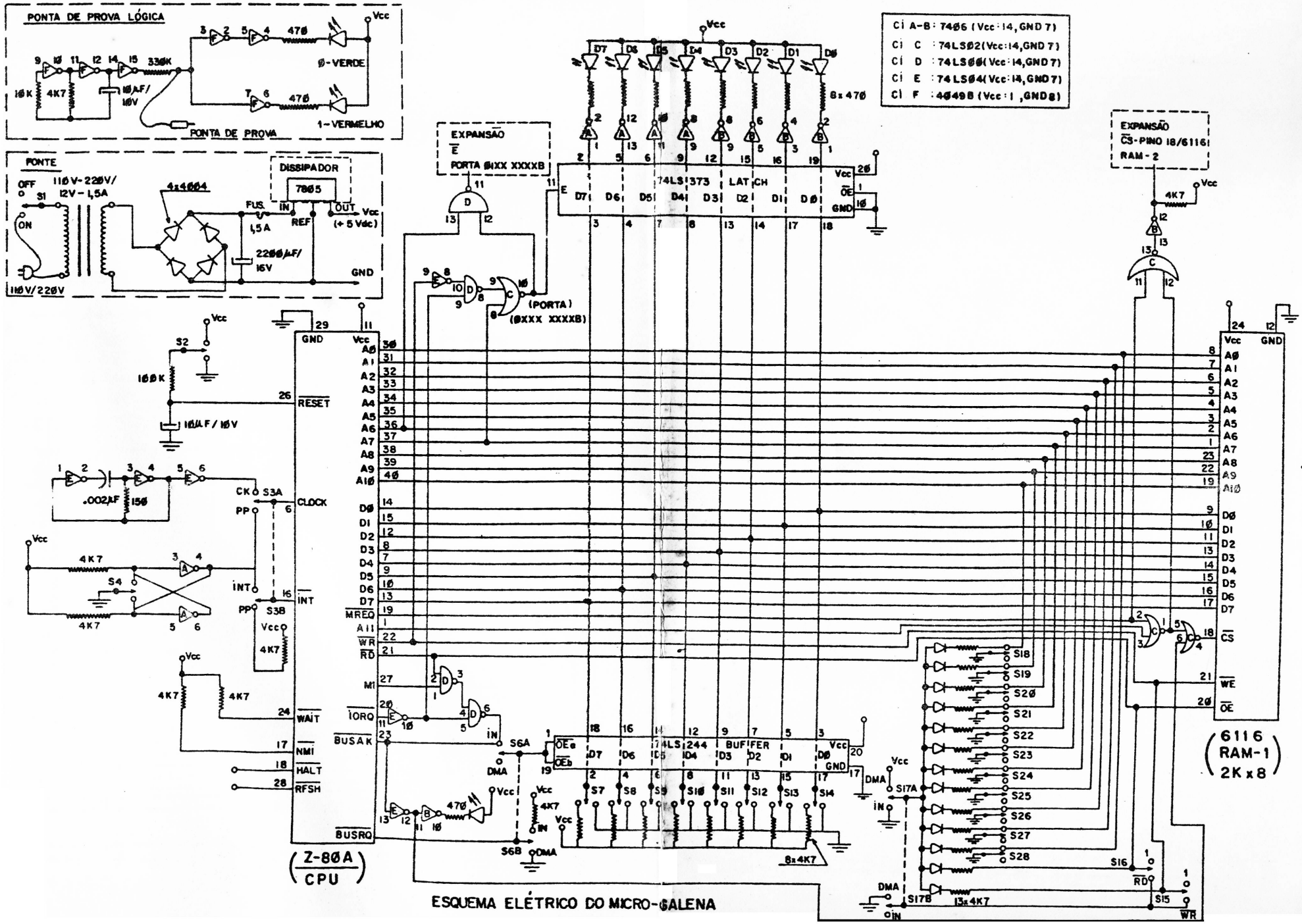
- NÃO AFETADA
. 1 OU 0 CONFORME RESULTADO DA INSTRUÇÃO
x ESTADO ALEATÓRIO



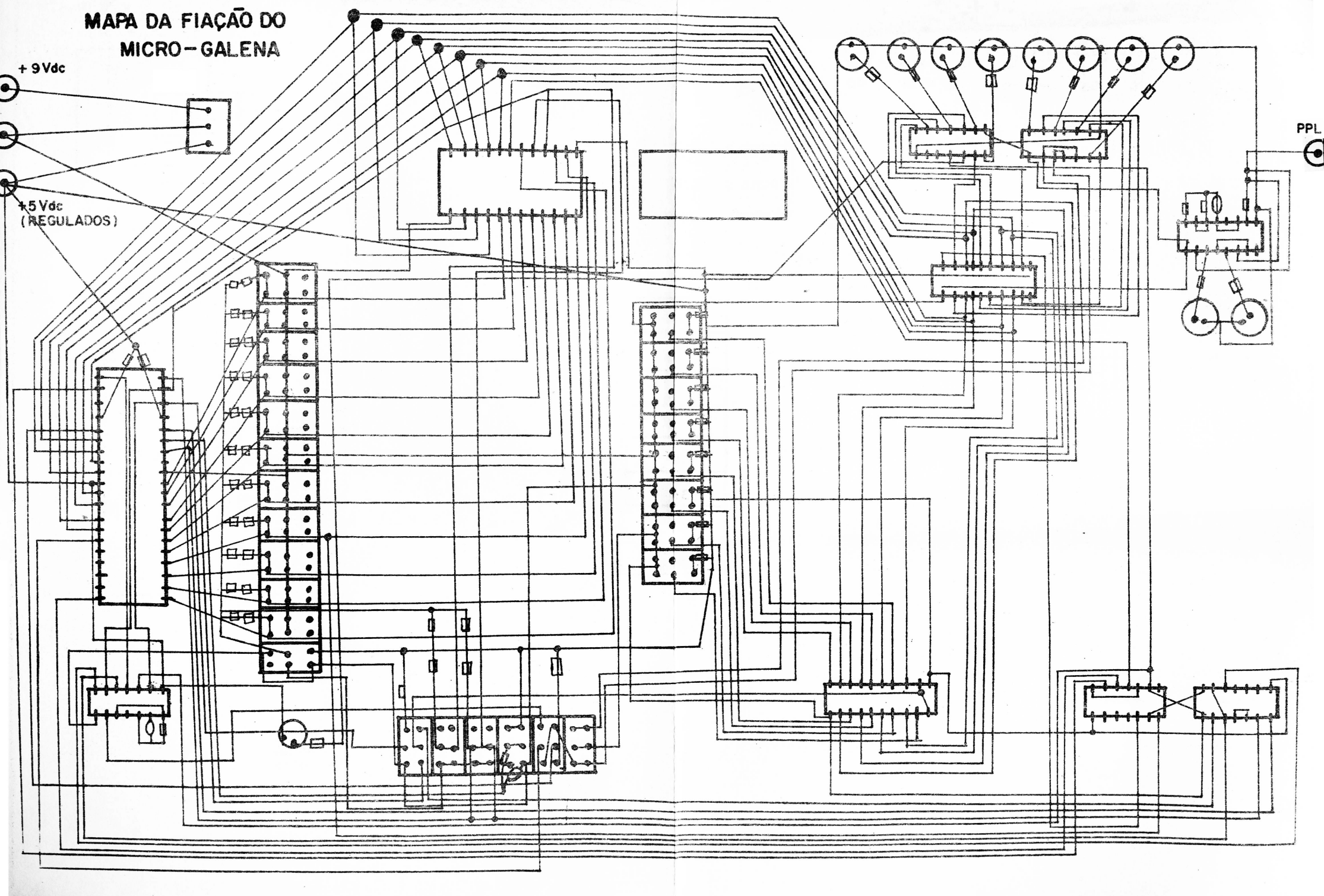
7-9 – Fonte de alimentação



CONSOLE DO MICRO-GALENA


PONTA DE PROVA LÓGICA
Vcc
330K
10K
4K7
10µF/ 10V
470
Ø-VERDE
1-VERMELHO
PONTA DE PROVA
PONTE
OFF
ON
S1
110V-220V/ 12V-1,5A
110V/220V
4x4004
DISSIPADOR
7805
FUS. 1,5A
IN
REF
OUT
Vcc (+5Vdc)
2200µF/ 16V
GND
CI A-B : 7406 (Vcc:14, GND 7)
CI C : 74LS02 (Vcc:14, GND 7)
CI D : 74LS00 (Vcc:14, GND 7)
CI E : 74LS04 (Vcc:14, GND 7)
CI F : 4049B (Vcc: 1 , GND 8)
8x470
EXPANSÃO
E
PORTA ØXX XXXXB
74LS373 LATCH
(PORTA)
(ØXXX XXXXB)
EXPANSÃO
CS-PINO 18/6116
RAM-2
100K
10µF/ 10V
.002µF
150
S2
S3A
S3B
S4
CLOCK
RESET
INT
WAIT
NMI
HALT
RFSH
BUSAK
BUSRQ
MREQ
IORQ
RD
WR
M1
A11
DMA
S6A
S6B
74LS244 BUFFER
8x4K7
13x4K7
6116 RAM-1 2K x 8
CS
WE
OE
Z-80A CPU
ESQUEMA ELÉTRICO DO MICRO-GALENA

MAPA DA FIAÇÃO DO MICRO-GALENA



COLA

COLA

COLA

LIVROS EM PROMOÇÃO

A PETIT EDITORA sempre está lançando novos livros, e a primeira tiragem é feita em forma de promoção junto aos nossos clientes cadastrados, através de mala direta com descontos especiais. Caso você não tenha seu nome em nosso cadastro mas gostaria de receber nossos folhetos com promoções, envie-nos o cupom abaixo preenchido, hoje mesmo! Pode estar saindo um livro novinho.

DOBRA

DESEJO TER MEU NOME NO CADASTRO DA PETIT EDITORA, PARA RECEBER OS FOLHETOS PROMOCIONAIS;

Nome _____

End. _____ **N.º** _____

Apto. _____ **Bairro** _____

Cep: _____ **Cidade:** _____

Est.: _____

Profissão: _____ **Fone:** _____

_____

Idade: _____ _____

Assinatura

TEU LIVRO

A existência na Terra é um livro que estás escrevendo...

Cada dia é uma página...

Cada hora é uma afirmação de tua personalidade, através das pessoas e das situações que te buscam.

Não menosprezes o ensejo de criar uma epopéia de amor em torno de teu nome.

As boas obras são frases de luz que endereças à Humanidade inteira.

Em cada resposta aos outros, em cada gesto para com os semelhantes, em cada manifestação dos teus pontos de vista e em cada demonstração de tua alma grafas com tinta perene, a história de tua passagem.

Nas impressões que produzes, ergue-se o livro dos teus testemunhos.

A morte é a grande colecionadora que recolherá as folhas esparsas de tua biografia, gravada por ti mesmo, nas vidas que te rodeiam.

Não desprezes, assim, a companhia da indulgência, através da senda que o Senhor te deu a trilhar.

Faze uma área de amor ao redor do próprio coração, porque só amor é suficientemente forte e sábio para orientar-te a escritura individual, convertendo-a em compêndio de auxílio e esperança para quantos te seguem os passos.

Vive, pois, com Jesus, na intimidade do coração, não te afastes d'Ele em tuas ações de cada dia e o livro de tua vida converter-se-á num poema de felicidade e num tesouro de bênçãos.

EMMANUEL

(Página recebida pelo médium Francisco Cândido Xavier)

ELETRÔNICA DE VIDEOGAMES – TEORIA E MANUTENÇÃO

Este é um livro destinado a técnicos e hobbistas interessados em conhecer a fundo essas fascinantes máquinas. Desenvolve toda a teoria, desde Eletrônica Digital até a técnica de geração de imagens no receptor de televisão, apresentando também as bases para a programação e cópia de cartuchos.

Os videogames ATARI e ODISSEY são analisados detalhadamente como exemplos, registrando-se todos os seus circuitos e discutindo-se a teoria e operação dos estágios.

Um capítulo especial trata de instrumentos e técnicas de consertos, inclusive a conversão NTSC-PAL, possibilitando aos técnicos e oficinas entrar nesse rendoso e crescente mercado.

CURSO DINÂMICO DE FOTOGRAFIA

Aprenda a fotografar, revelar filmes e fazer cópias em papéis sem utilizar qualquer equipamento especial, uma oportunidade rara para você conhecer os segredos da fotografia.

TELEVISÃO – Teoria e Consertos - Cores/PB

Televisão era um segredo desvendado apenas por meia dúzia de cérebros ilustres; estranhos esquemas e fórmulas que era aprendidas misteriosamente por eles. Consertar a Televisão era pura mágica; "macetes" e "jeitinhos" que vinham do berço.

"TELEVISÃO - TEORIA E CONSERTOS" é uma obra para acabar com esses preconceitos, apresentando uma visão moderna e aberta sobre o assunto. O "segredo" é reduzido a uma teoria bem ordenada e fundamentada, conduzindo o leitor pela lógica do aparelho. "Macetes" e "Jeitinhos" são transformados em técnicas de aplicação universal, acessíveis a todos que tenham dedicação, esforço e disciplina.

Na primeira parte do livro é desenvolvida a teoria de transmissão e recepção de TV, a cores e preto e branco, analisando-se os circuitos de cada estágio do receptor, inclusive controle remoto. Na segunda parte são apresentados instrumentos para consertos e ajustes, roteiros de ajustes e técnicas de consertos, finalizando com um extenso guia de defeitos.

O esquema do livro é válido para estudantes que desejam dominar essa área indispensável, para interessados em entrar no negócio de assistência técnica e Técnicos experientes precisando aprofundar seus conhecimentos.

CONSTRUA SEU COMPUTADOR POR MEIO SALÁRIO-MINIMO

O livro é direcionado a engenheiros, técnicos e estudantes interessados num aparelho de baixo custo para prática de Eletrônica Digital, microprocessadores e programação assembler/código de máquina.

O computador proposto no livro — MICRO-GALENA — pode ser construído no fundo do quintal, sem circuito impresso e utilizando ferramentas caseiras comuns. Todos seus componentes são facilmente encontrados no comércio nacional, inclusive pelo reembolso postal, pelo custo de meio salário-mínimo.

E nas horas vagas o MICRO-GALENA estará disponível para um joguinho.

MANUTENÇÃO DE MICROCOMPUTADOR

Quase 500.000 microcomputadores estão implantados no Brasil. O que fazer para que continuem funcionando? A resposta é desenvolver urgente e rapidamente um contingente de mão-de-obra especializada em manutenção, capaz de fazer frente ao desafio da vertiginosa produção de micros.

A PETIT EDITORA presta sua colaboração e lança a 3.ª edição de "MANUTENÇÃO DE MICROCOMPUTADORES", com detalhada descrição de técnicas, teorias e instrumentos envolvidos na manutenção de microcomputadores.

Os primeiros capítulos apresentam as bases teóricas — Eletrônica Digital, Microprocessadores e Arquitetura de Microcomputadores — incluindo um estudo sobre os microprocessadores Z-80, 6502, 68.000 e guia dos micros TK, CP e APPLE.

A redação e organização do livro é adequada para técnicos e engenheiros interessados em questões práticas de manutenção de microcomputadores, além de fonte de referência didática e consulta para estudantes que se iniciam na área.

RÁDIO — TEORIA E TÉCNICAS DE CONSERTOS

A teoria de rádio é desenvolvida de uma maneira fácil e acessível, dos princípios elementares até as aplicações mais complexas. Paralelamente à descrição dos estágios do receptor são apresentadas as técnicas de diagnóstico do efeito e como consertá-lo. A abordagem inclui sistemas de transmissão AM, FM, HI-FI, estéreo e os respectivos circuitos no receptor. O livro é recomendado para assistência técnica e aficcionados no mundo do rádio.

CURSO DINÂMICO DE SILK-SCREEN

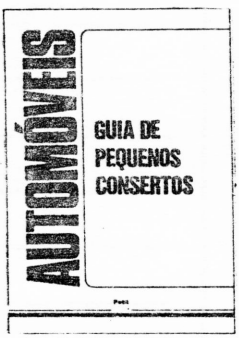
O que é Silk-Screen? Podemos chamar de um processo gráfico, através do qual é possível realizar estampas sobre as mais variadas superfície e nos mais diversos materiais; madeira, tecido, papel, etc. Mas se você pensa que só pode ser feito com máquinas, está muito enganado, muitas pessoas já estão se saindo muito bem e ganhando muito com esta profissão, trabalhando apenas no processo manual. De que maneira? Bem, isso você verá quando tiver o curso em suas mãos, estudar, praticar e por mãos a obra. Aí você poderá fazer estampa em camisetas, circuítos impressos, adesivos, etc. E poderá fazer tudo isso, somente nas suas horas livres, ganhando duas vezes, ou se preferir, dedicar-se exclusivamente à nova profissão.

ELETRÔNICA BÁSICA
TEORIA E PRÁTICA

Este é o caminho mais curto para se chegar ao reino da Eletrônica. Destacando o que há de realmente importante na área e mostrando os mínimos detalhes toda a técnica usada pelos veteranos de oficina e laboratórios, o livro habilita o leitor a atingir pontos mais avançados como rádio, TV e computadores.

AUTOMÓVEIS — GUIA DE CONSERTOS

Finalmente chegou o braço direito do motorista. Com este guia o motorista já não terá tanta necessidade de levar o seu carro ao mecânico quando surgir algum problema, basta pegar o seu guia, abrir na página certa e lá estará a solução, de uma forma muito simples e até simpática, amenizando até aquele nervosinho que as vezes passamos quando não conseguimos identificar a solução para o problema que está ocorrendo.
E sem falar na economia que o motorista terá com mecânicos, pois muitos problemas parecem grandes, mas que às vezes basta um pequeno massete para resolver, economizando no dinheiro e no tempo. Os defeitos são apresentados em ordem alfabética para uma melhor consulta, e as soluções são fáceis de assimilar, para qualquer marca de carro, a álcool ou a gasolina. Ideal para dar de presente à uma pessoa que gostamos, ideal para ter no porta luvas, para os momentos inesperados.



ELETRÔNICA DIGITAL – TEORIA E APLICAÇÃO

A experiência indica que a melhor hora de entrar num negócio é quando ele está iniciando. Computadores; robôs, telecomunicações, instrumentos — a Era Digital está começando. Surge a necessidade de técnicos especializados e de conhecimentos sobre a nova tecnologia. Para o técnico brasileiro é a oportunidade profissional numa área que é reconhecida como o futuro da humanidade.

Bits, bytes, microprocessadores, memórias, todo o mundo digital estará à sua disposição em "ELETRÔNICA DIGITAL – TEORIA E APLICAÇÕES" de uma maneira fácil e acessível, com minuciosos detalhes de tudo o que é importante, mas sem complicações desnecessárias.

Com "ELETRÔNICA DIGITAL – TEORIA E APLICAÇÕES" você vai entrar no novíssimo mundo da Era Digital e conhecer todo o instrumental teórico e prático que um profissional da área precisa. Sem mistérios e sem complicações.

Prof. Maurício Caruzo Reis

REMETENTE

Nome: _____

Endereço: _____

Bairro: _____

Cidade: _____ **Est.:** _____

Cep: _____

SELO

LOJAS DO LIVRO ELETRÔNICO

Caixa Postal 1131

RIO DE JANEIRO - RJ - Brasil

20001

OUTROS LIVROS DA PETIT

- **MANUTENÇÃO DE MICROCOMPUTADORES**
- **ELETRÔNICA DE VIDEOGAMES**
  Teoria e Manutenção
- **ELETRÔNICA DIGITAL**
  Teoria e Aplicação
- **ELETRÔNICA BÁSICA**
  Teoria e Prática
- **RÁDIO**
  Teoria e Técnicas de Consertos
- **SILK SCREEN**
  Para Eletrônica
- **TELEVISÃO**
  Teoria e Consertos - Cores/PB
- **AUTOMÓVEIS**
  Guia de Pequenos Consertos
- **FOTOGRAFIA**

PETIT
EDITORA E MARKETING DIRETO LTDA.

Av. Brigadeiro Luiz Antonio, 383 - SP CEP 01317